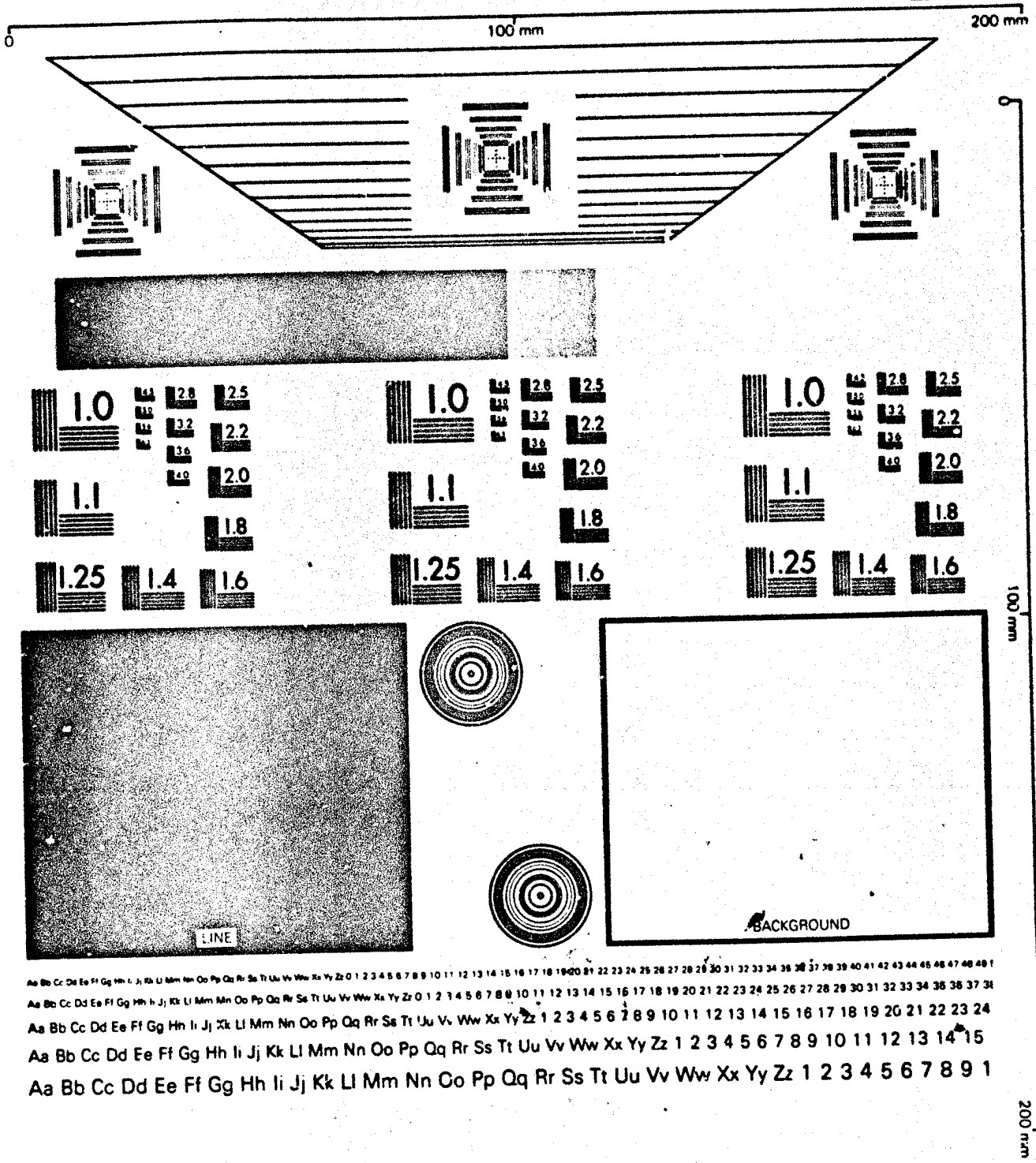
PROCEEDINGS OF
THE SYMPOSIUM/WORKSHOP
ON
APPLICATIONS OF EXPERT SYSTEMS
IN DND

3

ROYAL MILITARY COLLEGE OF CANADA
KINGSTON

CONTROL TEST TARGET 2.4x

Kodak Quality Monitoring Program



Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 5

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 1

Sensitized Imaging Products
Business Imaging Systems Division

7/87

A-2289

PROCEEDINGS
4th SYMPOSIUM / WORKSHOP
ON
APPLICATIONS OF EXPERT SYSTEMS IN DND

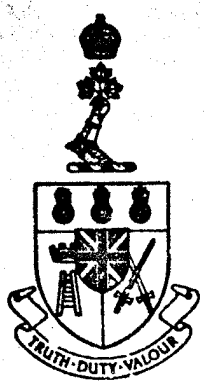
23–24 April 1992

ROYAL MILITARY COLLEGE OF CANADA
KINGSTON

4TH SYMPOSIUM / WORKSHOP

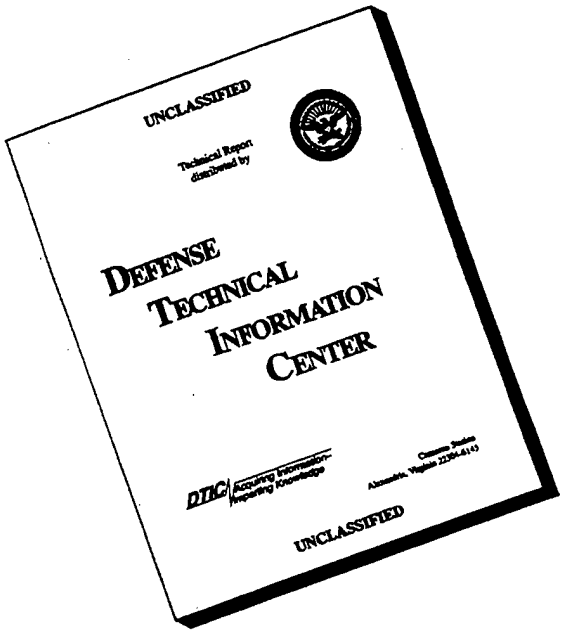
APPLICATIONS OF EXPERT SYSTEMS IN DND

RMC, KINGSTON



APRIL 1992

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

PROCEEDINGS
4th SYMPOSIUM / WORKSHOP
ON
APPLICATIONS OF EXPERT SYSTEMS IN DND

23–24 April 1992

ROYAL MILITARY COLLEGE OF CANADA
KINGSTON

EDITORS:
Dr. Georges AKHRAS
Civil Engineering
Dr. Pierre ROBERGE
Chemistry & Chemical Engineering

PREFACE

The engineering community is presently going through a major transformation brought about by the miniaturization of computer systems. One of the important aspects of this transformation is that most of these systems deal directly with knowledge - from data manipulation to specifying recommendations. Expert Systems, a successful by-product of Artificial Intelligence, are good examples of these knowledge handling tools. Affordable hardware and software make Expert System technology easily accessible to the most remote users of expertise. The fact that operative knowledge can now be transported onto diskettes and updated on-site brings a renewed flexibility to various domains of engineering. The constant drain of corporate memory and knowledge imposed by postings and transfers in DND could be minimized by capturing and storing expertise in these systems.

The first Symposium/Workshop on the Applications of Expert Systems in DND was held in March 1989 at RMC and many of the participants expressed the desire to hold regular, similar meetings in the future. Since then, a Symposium/Workshop has been organized each year.

As for the previous Symposium/Workshops, this one is organized with the intent to create a vehicle, within the Department of National Defence, to promote, stimulate and demystify the development of Expert Systems and their applications to engineering. The material and information contained in the papers are published in the exact form as given by the authors. Any statement, opinion or view expressed in the papers are the sole responsibility of the authors. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

We would like to express our sincere appreciation to the authors for their contribution and cooperation in the production of these proceedings. We are grateful to the staff of the Royal Military College of Canada for their support in organizing this meeting.

The editors.

G. Akhras and P. R. Roberge

TABLE OF CONTENTS

TABLE OF CONTENTS

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

DEVISING A BATTLE PLAN WITH CASE-BASED REASONING

James Bowen[1]

Abstract

When devising strategic and tactical plans an important aspect to consider is the response of the opposing force for the implementation of the plans. If a computerized system could be build which would anticipate opposing force reaction to offensive movements it could result in discovering weaknesses in strategic and tactical plans. This system would need to take into account many different pieces of information in order to determine possible opposing force response tactics.

One such method of generating possible responses must first determine if the opposing force commanders learn by examining previous military situations. This method of reasoning was observed in the recent war when defense strategies used resembled those of WW I.

If so then a method called case-based reasoning might provide the basis for such a computerized system. This system could combine limitations imposed by the environment, eg. personnel, orders, equipment, etc., with previous military situations to suggest responses to planned offense movements. This paper discusses the appropriate military situation, case-based reasoning, previous applications, a military example, limitations, the research issues, and a brief comparison to rules and neural nets. Finally, a case-based reasoning system could be the beginning of a true learning machine which would continually increase its understanding of opposing forces.

[1]CompEngServ Ltd., Suite 300, 19 Fairmont Ave., Ottawa, Ontario, K1Y 1X4 (613)722-3008

MILITARY REASONING

In general, a military commander's strategic and tactical response to events generated by the opposing force is guided by six considerations: direct orders, military policy (e.g. guidelines, contingency plans, etc.), intelligence/information, experience/training, limitations of equipment/resources/terrain/personnel/etc., and analysis of previous military situations. As a case in point, Iraq's strategical and tactical manoeuvres with Iran and later with coalition forces were reminisced of WW I defense strategies.In addition, coalition ground strategies were reminiscent of the Korea war and WW II.

At the battlefield level, in totalitian regimes such as communist states (USSR ?) or dictatorships (Iraq) there is little room for creative or free thinking on the part of the commander. If faced with a situation outside the direct orders or known policy, response might be determined by using solutions proved successfully in similar previous situations. This also allows the commander to justify action taken at a later date.

If a system could be designed which would incorporate known military policy, restraints on equipment, resource, terrain, personnel, etc., as well as, previous military situations it could have potential in determining response to offensive movements. This could be done by determining a profile of the capabilities of the opposing side and its studied or experienced military situations. Each previous military situation could be compared to anticipated offensive movements and the closest match may indicate opposing force response. Such a system could indicate a weakness in offensive movements.

In the case of "freer" nations a wider definition of "closest match" could be used.

In addition, such a system could incorporate the individual profile of a Canadian Commander's experience to produce a system to probe for weaknesses in their thinking. If properly designed it could be used in a war games approach. Where the system reacts to the user's movements according its role as the opposing force.

Recently, a need has been expressed for computer support in devising plans for a tactical and mission oriented pilot's associate; autonomous land vehicles; naval battle management; and, the air land battle concept of NATO [1] [2]. This paper is designed to give an overview of a technology, which may be appropriate for such systems, called case-based reasoning.

CASE-BASED REASONING

Artificial Intelligence (AI) encompasses many different areas such as natural language processing, robotics, etc [3]. One area is machine learning (although some argue it is not a part of AI). In machine learning there are 4 main subareas: Induction (learning a general concept from viewing both positive and negative examples), analytic (deductive approaches using past experiences to provide examples and guide the solution for a new problem, for example, case-based, analogical, explanation based learning), genetic algorithms (produce mutations of examples by selecting the best features of a class and use a selection process to achieve objective) and connectionism or neural nets.

Before discussing Case-based learning its worth while to discuss a few points about learning in general, first some definitions [4]:

1) Analogy which infers resemblance between two objects. For example, class attributes of a first case are listed and a second case is compared to see if it is a member of the same class of cases;
2) Generalization may occur when two or more cases share two characteristics and where a class of additional cases share one of the

characteristics. In the generalization process the second characteristic is inferred as an attribute of the additional cases as well.
3) Cause and ... ct connections can be inferred between events or attributes which ... pear to occur together for cause and effect analysis.

In general, learning approaches may require the following abilities [4]:

1) The ability to reason with cases and examples through analogy;
2) Handle ill-defined, open textured problems i.e. to prove an argue through a logical analysis of subjective information with little or no predefining parameters;
3) Handle fundamental conflicts between previously generated rules and newly generated rules;
4) Handle change and non-monotonocity, which is defined as the constant change in the proposed rules as the underlying cases change.

If a computerized learning system was built it would have [4]:

1) Indexing abilities in order to search for relevant previous cases;
2) Assessment abilities of similarities and differences between the current case and previous ones;
3) The ability to generalize in order to evaluate potential solutions;
4) The ability to adapt to changing underlying facts.

Case based reasoning (CBR) has received some attention lately as theoretic concepts are now being applied to real problems [5] [4]. This type of reasoning got its start with the theory of Dynamic Memory Structures [6] [7]. Case-based reasoning can be defined as the ability to match the facts of a situation against previous ones and adapt the solution to the new case. The central idea behind CBR is that reasoning is done from previous cases and not from "first principles", i.e. conclusions are drawn from the available case. This allows a problem solving and reasoning system to avoid known errors and reach a solution in a more guided way, i.e. incorporating the solution methodologies found successful in previous cases. The basic method is to first select a

case(s) which best matches the current situation. Next, these cases are slightly changed or even drastically modified in order to more closely match and solve the current case. If the solution is correct the new case may be added to the knowledge base.

PREVIOUS CASE-BASED REASONING SYSTEMS

Now, with an idea of general learning concepts and what case-based reasoning is, the following section examines previous computerized uses of it. This learning approach has been used in a variety of areas. For example, Chess has been a popular application area [8] [9], understanding human intentions in military situations [10], accomplishing Natural Language processing [11]. Some authors have suggested a planner that would retrieve and modify past experiences as guides to solving new cases. Their work was concerned with adversarial planning.

ARIES [12] used a certain type of memory for the storage of cases and a partial matcher with a built-in criterion and an indexing mechanism for case retrieval. The retrieved case was changed heuristically until it satisfied the requirements of the current problem. Because of the obvious problems of such a retrieval strategy (ARIES cannot consider alternatives or retrieve subproblems, instead of whole problems, from memory). The author presented a new methodology and a novel definition of when two cases are "similar": two problems are similar if their initial analysis yields the same reasoning steps, i.e., they consider the same issues and make the same decisions [13]. Furthermore, by storing alternatives, failed solutions and reasons, the system is capable of better utilizing its previous experiences.

MEDIATOR is a case-based system that stores its cases as generalized episodes [14] [15]. MEDIATOR uses an *a priori* evaluation of the closeness of fit of a plan in memory to the current case for old plan selection. Indexing is furthermore enhanced by the features that describe possible plan failures.

WOK [16] and CHEF [17] [18] are two case-based planning programs that work in the domain of cooking. WOK generates plans by retrieving an old one from memory and modifying them to fit the current problem. WOK is unique in its retrieval method, since it uses subgoal interactions as indexing mechanism. CHEF retrieves plans according to the number of goals they satisfy and the interaction of goals, and then modifies them to satisfy unsatisfied goals.

PLEXUS is a planner applied to common-sense reasoning [19]. PLEXUS retrieves old plans by use of the background knowledge of a plan that includes general plans, categorization and causal knowledge.

Finally, TOLTEC is a case-based planning system that has been applied to manufacturing problems [20].

The domain of application of CBR systems is constantly being expanded. New systems are attempting to apply CBR in legal reasoning, battlefield management, medical diagnosis, manufacturing, and so forth. CBR is viewed by many as the new methodology for intelligent systems, which will eventually replace the current rule-based formalisms for certain types of problems.

SAMPLE MILITARY CBR

The following very simple example suggests one of many possible approaches that could be looked at in a military CBR study. In order to construct such a system the following methodology might be used:

1) Identify the features in each previous military situation/case. Assuming each case contains both information of general content and technical expertise. The technical expertise might be extracted by: a) Identifying key words, phases or combinations of letters which represent the possible inputs. This could cover weapons, geographic, tactics, strategic intent, etc. b) In the situation where a tactic can be linked in a hierarchy.

The next step is to identify second level or as many levels as necessary of resulting tactics. This could occur where there is more than one tactical goal. c) Establish relationships between the features, establish strength and direction of relationship, i.e. causality. For example, troop and weapons movements. d) Identify results or causes of features and establish relation between features and results/causes. For example, troop movements may precede road and radar upgrades or aggressive political statements. e) Identify and establish relationships between results and successful counter offense/defense. For example, static defense lines may be by-passable. f) This system would have to be able to distinguish and deal with commonly accepted words such as "airborne" and variable standard words such as adjectives like "massive".

2) A software tool to translate cases into structured representation, the features, the military situation and possible responses must be linked. This relationship would contain three values: the link, causality, degree of strength. The main result would be the ability to link each feature with its defensive manoeuvre and give a certainty value.

3) Once operational, a machine learning system would accumulate new rules from new cases. This system would have to address the issue that new cases might either change the relationship derived earlier or extract new relationships. This system would have to be able to handle conflicts generated by new cases. For example, in the situation where several tactical defensive solutions are acceptable given the same offense tactics.

4) The resulting symptom would then be able to recognize the key features of a new case, match them against previous cases and derive a listing of probable causes. Since each feature has a degree of strength associated with it and a cause, the system could compute, using an appropriate certainty or probability scheme, which cause(s) are most probable. At this point, the same kind of procedure could compute the listing of most probable defensive tactic.

PRIMACY OF RECENT DATA

The following two sections discuss heuristics for designing such systems.

One important point that has been learned is: recent data would always take precedence over older data [21].

The best way to make certain the most recent data in the system takes precedence over old data is to run cases from the data base through the learning system in chronological order (when training the system). This would be necessary as advancements in military technology continually allows more options. Using this form of the algorithm, the system does not need to search the data base. Instead, if the system generates its own rules derived from cases then for each new case, it searches its own rule base until it finds the rule that best matches the case. The rule it finds might be based on a single case or could be a generalization from many cases. Then the system needs to check the conclusion of the rule and compare it to the correct decision on the case it is considering. If the decisions match, the system can create a more general rule (or just leave the current rule in place).

However, if the decisions differ the system needs to create a new rule based on the new case. In addition, it needs to examine how closely the old rule matched the current cases. The system designers need to be able to vary this strength of match parameter. If the rule does not match very closely, the system can create a new rule that matches the current case better than the old rule.

If the old rule matches very closely the system must make a more sensitive decision. A close match means the system is looking at a case that is very similar to previous cases, but the correct answer is different from what it has been in the past. The new case might represent a change in policy or in the world e.g. technology, and the old rule is no longer appropriate. The system may then decide to replace the old rule with the new one.

MACHINE LEARNING LIMITATIONS

Despite its tremendous potential, machine learning cannot solve every problem [21]. It only works well for domains in which a large body of experience exists. Naturally, this experience should be available on line since the effort of encoding it could be enormous. Learning systems also might not work well for real-time applications since they have to search a large rule base to find the best rules for each new situation.

A more significant point to keep in mind is that 'learning' systems cannot decide for themselves what parts of the data are important. If the military had a very rich data base with lots of data for each case, the system engineers must first decide which information in each case is most important. This filtering of data almost always requires the assistances of experts, so a modified knowledge engineering role is still part of the process of building a learning system. If the data is fed into the program unfiltered, the contradictions or incompleteness, etc. in cases might be so great (for example, in the form of spurious correlation, which would lead to rules that could never work) that the program might never adequately learn.

Finally, systems that learn from experience cannot learn a principle that is not embodied in the experience they've had. If military intelligence senses the opposing force is operating on a different principle such as conserving manpower or concentrating on air defences and wants the learning system to make decisions to reflect that change, the engineers responsible for the system may need to modify it manually.

RESEARCH ISSUES IN CASE-BASED REASONING

The following section discusses research issues for a CBR system [4].

The typical reasoning of a CBR system involves selection of the appropriate case in memory, modification of the old case according to the new problem, and possible storage of the new solution as a case in memory. Each step of this cycle has many open research issues associated with it.

First, the question of how to represent the case and its features in computerized form must be addressed [22].

Second, a CBR system must select the best case or cases from memory. The question that must be answered is what constitutes an appropriate case. What are the criteria of closeness or similarity between cases, and how should cases be indexed. In the simplest approach, find which case has the most matching features. An important issue is to determine if the features are of the same important [22].

Third, a new solution must be checked to decide if it solves the problem. Checking a solution can take many forms. Some situations allow the application of analysis techniques to determine the goodness of a solutions; other situations may require simulation; others may allow a formal proof of solution correctness. Whatever the means, after a solution is checked the results of this check must be evaluated. If the solution is acceptable, based on some domain criteria, then the CBR system is done with reasoning. Otherwise, the case must be modified again, and this time the modifications will be guided by the results of the evaluation of the solution.

Fourth, after an acceptable solution has been found, the CBR system must decide if and how it should be added to its dynamic memory. During this learning phase the CBR system must decide if the new solution is sufficiently different from others, already known ones, to warrant storage. If it does warrant storage, it must be decided how the new case will be indexed, on which level of abstraction it will be saved, and where it will be put inside the dynamic memory organization.

Indexing a case is essential in establishing similarity, since the indices help define the elements of a problem that are important and which should be considered when studying a problem. Part of the case is the knowledge gained from solving the problem represented by the case. In other words, cases should also be indexed by some elements of the solution. According to the theory of dynamic memory the events that help most in reminding are failures: we remember situations where our expectations failed easier than situations which were stereotypical and according to expectation. Thus, any indexing scheme should include failures and deviations from stereotypical events. Two main approaches exist: First, hierarchical approaches arranged features from general to specific i.e. a decision tree. This is most suited for domains using well defined searches [22]. The associative approach involves linking features independently this allows more flexible searches.

BRIEF COMPARISONS

How does CBR compare to other AI approaches [22]. In comparison to rules, development time is spent creating features from cases rather than trying to articulate an expert's heuristics. In addition, a CBR provides better explanations of decisions by direct links to previous decisions rather than through a "distilled" rule base. This is also a problem with neural nets which can not provide explanations for their classifications of new problems. Once the system has generated a set of strategic or tactical plans, experienced personnel could then enhance it by devising creative and unexpected modifications. Thus, receiving explanation of how the plan was developed is extremely important in assessing its strengths and for use in extending or refining the computer generated plan.

CURRENT STATUS OF CBR SOFTWARE

Currently there appear to only be two publicly available CBR software systems, they are relatively expensive (>$10,000 US) systems: Remind by Cognitive and CBR Express by Inference (see review in October 1991 issue of "AI Expert").

CONCLUSIONS

This paper described a methodology called case-based reasoning which could have potential in developing both strategic and tactical offensive and defensive plans. It summarized a list of research issues which must be addressed both a fully automated system could be developed. However, such an approach could be the foundation for generating battle plans which draw upon accumulated military experience.

REFERENCES

1. Solem E., "Military Uses of Expert Systems: Some Future Perspectives", Proceeding of the Symposium/Workshop on Applications of Expert Systems in DND, RMC, 1989, pp. 21-56.

2. Solem E., "Some Longer Term Technological Trends in Artificial Intelligence (AI) and Expert Systems (ES) - An Overview", Proceeding of the 2nd Symposium/Workshop on Applications of Expert Systems in DND, RMC, 1990, pp. 12-26.

3. Carbonell J., "Machine Learning: Paradigms and Methods", MIT Press, 1990.

4. Rissland E.L., "Artificial Intelligence and Legal Reasoning", AI Magazine, Vol. 9, No. 3, 1988, pp. 45-55.

5. Tsatsoulis C., "Case-Based Design and Learning in Telecommunications", Second International Conference on Industrial and Engineering

Applications of Artificial Intelligence & Expert Systems, ACM, 1989, pp. 509-518.

6. Ableson R.P. and Schank R., "Scripts, Plans, Goals and Understanding", Lawrence Eribaum Associates, Hillsdale, NJ, 1977.

7. Schank R., "Dynamic Memory: A Theory of Reminding and Learning in Computers and People", Cambridge University Press, 1982.

8. Wilkins D. E., "Using Plans in Chess, International Joint Conference in AI, 1979, p. 960-967.

9. Luck K. V. and Owsnicki B., "N.N. A View of Planning in Chess, 6th German Work Shop on Artificial Intelligence, Berlin: Springer Verlag, 1982, pp. 92 - 101.

10. Cross S., Banij R.B. and Norman D.O., "Knowledge Based Pilot Aids: A Case Study in Mission Planning", International Seminar by Deutsche Forschungs Und Versuchs Anstaltfuer Luft and Raumfahrt, H. Winter (Ed.) Berlin: Springer Verlag, 1986, pp. 141-174.

11. Schank R. and Childers P.G., "The Cognitive Computer", Addison Wesley, 1984.

12. Carbonell J., "Learning by Analogy: Formulating and Generalizing Plans from Past Experience", in Machine Learning: An Artificial Intelligence Approach, Tioga Press, Palo Alto, 1983.

13. Carbonell J., "Derivational Analogy and its Role in Problem Solving", AAAI-83, 1983, pp. 64 - 69.

14. Kolodner J. L. and Simpson R.L., "Experience and Problem Solving: A Framework", Proceedings of the 6th Annual Conference of the Cognitive Science Society, 1984, p. 2 - 9.

15. Simpson, R. L., "A Computer Model for Case-Based Reasoning in Problem Solving", Ph.D. Thesis, Report GIT-ICS-85/18, School of ICS, Georgia Institute of Technology, 1985.

16. Hammon K. J., "Planning and Goal Interaction: The Use of Past Solutions to Present Situations", AAAI-83, 1983, pp. 148 - 151.

17. Hammond K. J., "CHEF: A Model for Case Based Planning", AAAI-86,

1986, pp. 267 - 271.

18. Hammond K. J., "Learning to Anticipate and Avoid Planning Problems thorough the Explanation of Failures", AAAI-86, 1986, pp. 556 - 560.

19. Alterman R., "An Adaptive Planner", AAAI-86, pp. 65-69, 1986.

20. Tsatsoulis C. and Kashyap R.L., "A Case-Based System for Process Planning", International Journal of Robotics and Computer-Integrated Manufacturing, vol.4, no. 3/4, 1988, pp. 557 - 570.

21. Salzberg S., "Machine Learning Comes out of the Lab", AI Expert, No. 2, 1988, pp. 44-52.

22. Barletta R., "An Introduction to Case-Based Reasoning", AI Expert, Vol. 6, No. 8, Miller Freeman Publications, 1991, pp. 43 - 53.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

A DECISION SUPPORT SYSTEM FOR ELECTRONIC WARFARE ANALYSTS

Major J.G.L Dumouchel[1] and Dr A.L. Jenkins[2]

Abstract

One of the main goals of Electronic Warfare (EW) is to identify enemy units and determine their intentions. This is accomplished through the analysis of data obtained by Electronic Warfare Support Measures (ESM). One of the techniques employed by EW analysts is to use deployment templates as defined in doctrinal publications to predict the deployment and intentions of the enemy forces facing our own forces. This paper presents a prototype Decision Support System (DSS) written in the Smalltalk programming language. It makes use of doctrinal templates to assist the EW analysts in estimating enemy locations, identities and intentions. Initial reactions to the DSS suggest it is a tool that will greatly enhance the working environment of the EW analysts.

[1]Project Manager, Director Signals Engineering and Maintenance, NDHQ, Ottawa
[2]Professor, Dept of Engineering Management, RMC, Kingston

INTRODUCTION

EW, which is defined as military actions involving the use of electromagnetic energy to determine, exploit, reduce or prevent hostile use of the electromagnetic spectrum, and actions to retain its use by friendly forces, has been practised in every major conflict since radio communications were first used. EW, although an integral part of air and naval operations, was generally ignored by western armies after World War II until the Vietnam War. Today, increased efforts are being devoted to this critical component of combat power. Modern warfare is becoming increasingly dependant on high technology command and control, surveillance and weapon systems, the majority of which use some portion of the electromagnetic spectrum for guidance and/or communications. The side that best makes use of the electromagnetic spectrum and reduces the enemy's use of the same spectrum will have a decided advantage in winning the war.

ELECTRONIC WARFARE OPERATIONS

EW [1] embraces the following three divisions:

a. Electronic Warfare Support Measures (ESM);
b. Electronic Counter Measures (ECM); and
c. Electronic Counter Counter Measures (ECCM).

ECCM are the defensive measures taken to protect friendly electronic systems against the enemy threat. ECM are actions taken by friendly EW units to prevent or reduce the enemy's use of the electromagnetic spectrum. ESM involve actions taken to search for, intercept, locate, record and analyse radiated electromagnetic energy for the purpose of exploiting such radiations in support of military operations. ESM provide a source of EW information required to conduct ECM operations, ECCM planning, threat selection, warning and avoidance, target acquisition and warning. The divisions of EW and their relationship with other EW terms can be best illustrated as in Figure 1.

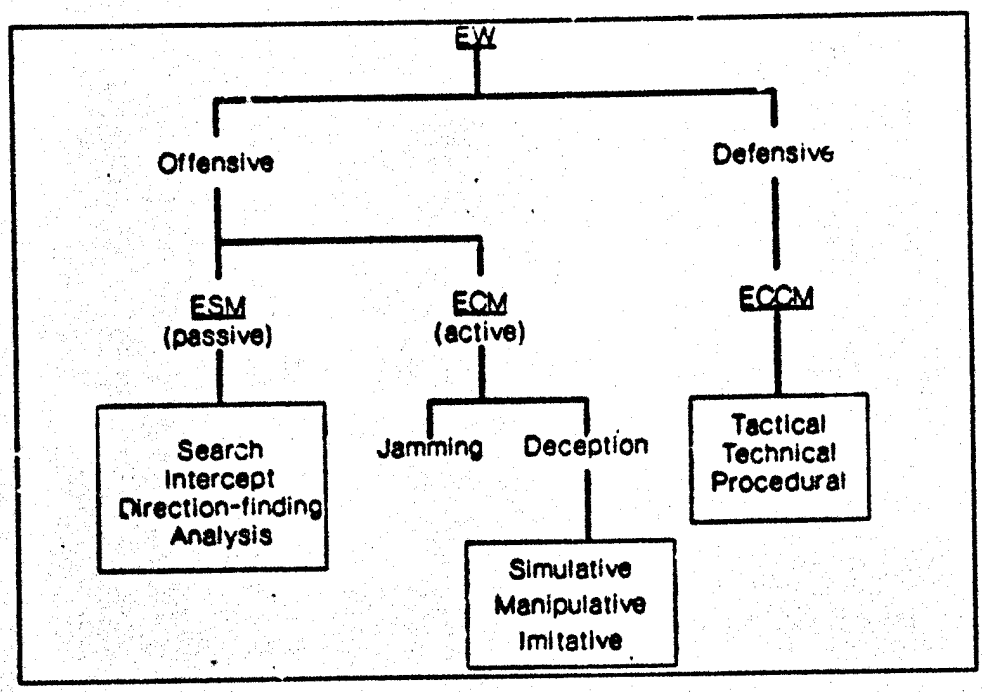


Figure 1 - EW family tree

THE NEED FOR AUTOMATED ANALYSIS

EW is an activity that has several inter-related and inter-dependant components. Figure 2 summarizes the entire EW process. The supported commander needs information about the enemy deployment and intentions to effectively deploy his own formations. One of the many sources for this information is the EW unit under his command. Therefore, the commander, through his staff, gives directions to the EW unit as to the type of information he requires to formulate his plan. This is known as the commander's Primary Information Requirements (PIRs) and Information Requirements (IRs). The EW unit operations centre translates these PIRs and IRs into tasks to be performed by the various detachments deployed by the unit.

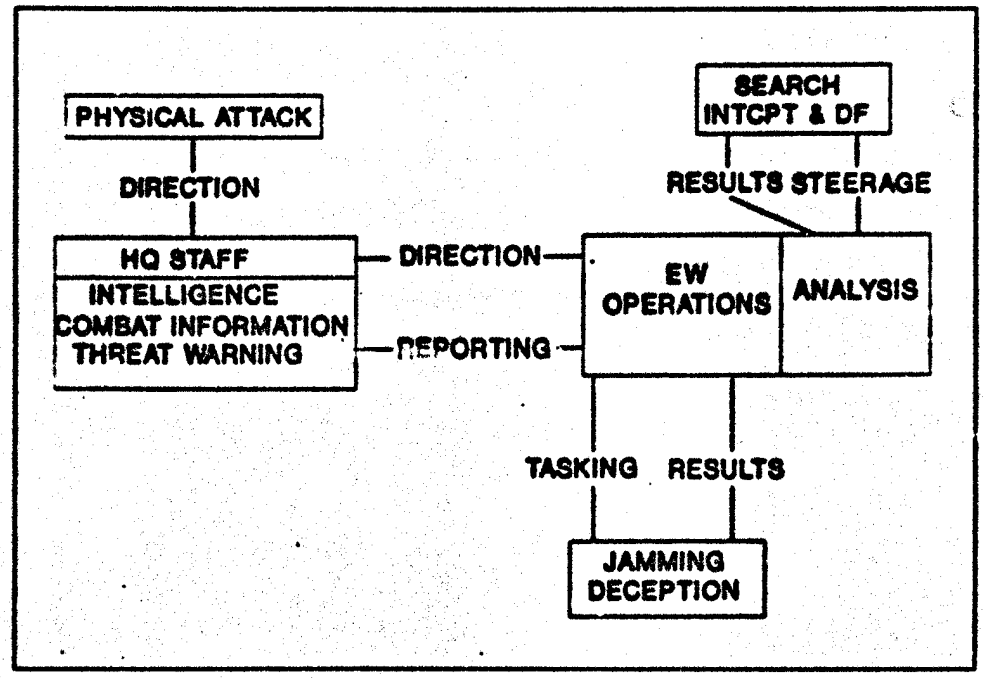


Figure 2 - EW process

The analysts will provide steerage to the search, intercept, direction-finding and jamming detachments in the form of:

a. known frequencies in use by the enemy;
b. types of signals used (e.g. voice, morse code or encryption);
c. known callsigns used on a specific radio network;
d. codewords commonly used on a particular radio network; and
e. the location of known emitters.

Using the data obtained from ESM, the EW analysts produce reports for the supported commander and his staff. These reports typically provide the following information:

a. tentative identification;
b. last known location;
c. predicted intentions;
d. approximate time when the action will take place; and
e. general comments from the analysts.

With the intelligence obtained from these reports and other sources of information, the commander makes decisions and gives directions to the units and formations under his command to either physically or electronically attack the enemy. In the case of ECM tasks (electronic attack), the analysts will provide the ECM controller with frequencies and locations of target emitters.

Analysis consists of scrutinizing all available information and extracting that which is important to the intelligence process. Figure 3 portrays the intelligence process.

From intercept, the analysts receive information about frequencies, callsigns, types of networks, message content, traffic flow, activity patterns and transmissions types. This information is further enhanced by information on locations and possible movement of emitters provided by direction-finding detachments. Other sources of information such as air reconnaissance, special forces, battlefield surveillance and weapon locating radars as well as prisoners of war add to the mass of data to be scrutinized.

The target array facing an EW unit can be huge and sophisticated. For example, it could be up to 175,000 emitters in the case of an army patterned along the lines of the old Warsaw Pact organization [6]. Obviously, all these emitters are not active at the same time. However, prior to the conduct of any operations, an EW unit will have gathered some information about the enemy emitters while building a database of information about the opponent force. The analysis of this information will result in reports to the G2 (Intelligence) and/or G3 (Operations) staff containing intelligence concerning the enemy order of battle (ORBAT), its strengths and intentions, and unit identities and equipment.

One can say that the EW analyst is a detective who quickly seizes upon any errors or breaches of security committed by the enemy. When the opposing force security measures are effective, the information obtained will be very fragmented and will only gain clarity over a long period of time. However, time is an advantage the analyst does not have. Accordingly, he needs some form of assistance to build up the enemy picture as fast as possible and as accurately as possible.

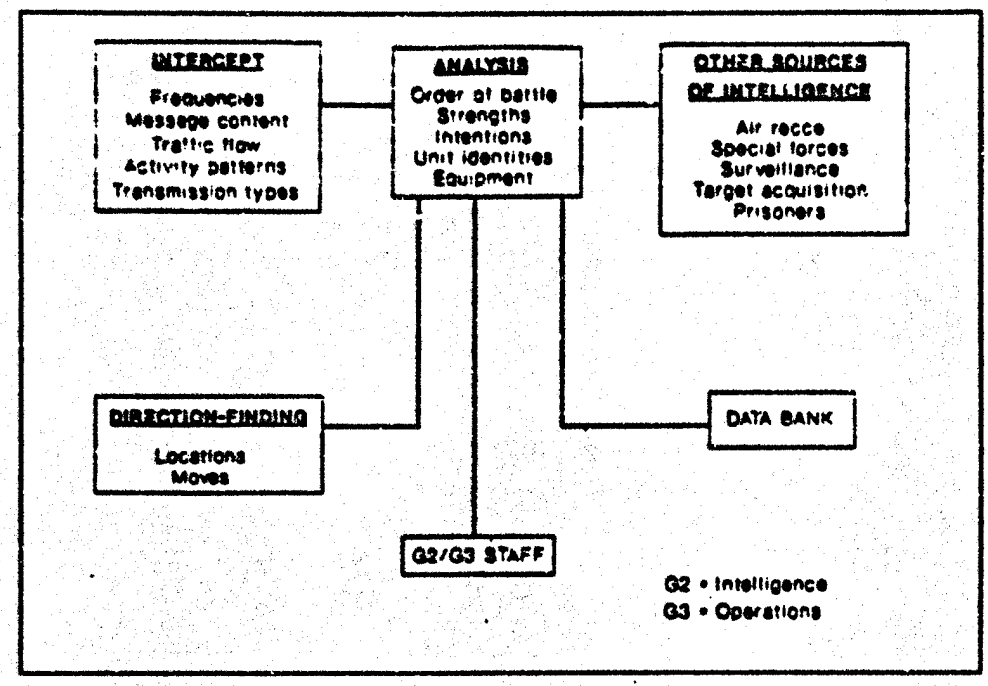


Figure 3 - The Intelligence process

Analysts use different techniques to evaluate the tactical information acquired through ESM and other sources. One of these techniques is called templating. It refers to the use of deployment templates as defined in the enemy's doctrinal publications to predict the deployment and intentions of the enemy forces facing our own forces. An example of such template is shown at Figure 4. These templates show the major elements of a formation or unit as well as their relative positions to each other and the Forward Edge of the Battle Area (FEBA). Commanders at all levels will normally comply to these templates while planning their deployment because doctrine usually affords them very limited latitude. Consequently, these templates are a very important tool for the analyst because it allows them to deduce:

a. enemy intentions. The type of template that best match the actual deployment will indicate the type of operations to expect from the enemy (e.g. attack, defence etc); and

b. <u>composition of enemy forces</u>. The template will prompt the analyst as to the major elements to expect to find in the opponent's unit or formation.

Most of the information gained from ESM can be easily handled by machines without the need for human intervention. It also appears that the use of templates will lend itself rather handily to automation. Consequently, designing a system capable of comparing the tactical data gathered through ESM and other sources to different doctrinal templates known to be usually used by the enemy will greatly enhance the analyst's ability to cope with the enormous amount of information available from the target array.

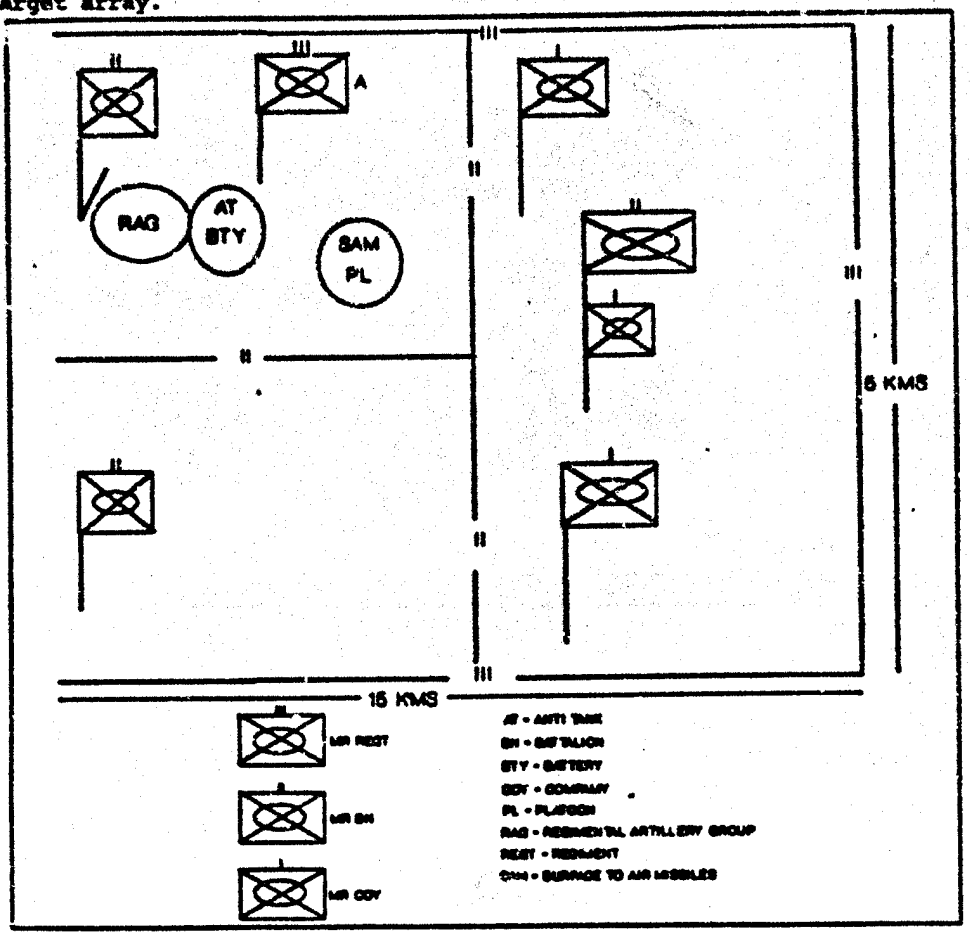


Figure 4. Typical doctrinal template

8

THE EW ANALYST SYSTEM

The EW Analyst System (EWAS) is a system that is designed to use doctrinal templates [2,6] to assist the EW analysts in the performance of their duties. It was programmed on a microcomputer using the Smalltalk [3,4,5,7] environment. The major features of the system are:

a. a window-based user interface;

b. an interface with a database of doctrinal templates to:
   1) add or delete categories of templates;
   2) add, delete or rename types of templates; and
   3) add, insert or delete an element in a given template;

c. an interface with a tactical database to:
   1) add, delete or rename units in the database;
   2) add or delete characteristics of a unit; and
   3) search a database of inferred (doctrinal) templates for possible matches to a tactical unit;

d. an interface with an inference calculator to:
   1) define an inferred (doctrinal) template;
   2) calculate the expected location of each element of the inferred template; and
   3) add the inferred template to the database of inferred templates;

e. an interface with a database of inferred templates to search the tactical database for a possible match to an element of the selected template or the selected template itself; and

f. the ability to compute a certainty factor for each match.

As the system requires doctrinal templates from which to compare the gathered information, a Template Database Browser is provided to the EW analyst to permit him to enter different doctrinal templates in a database. This browser is shown at Figure 5.

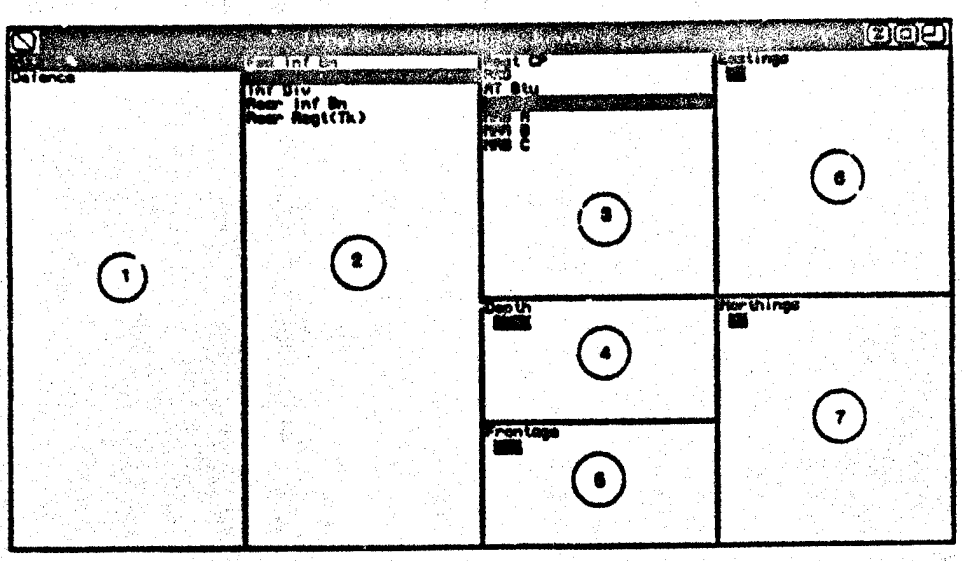


Figure 5 - Template Database Browser

This browser has seven (7) panes allowing the EW analyst to:

a. add, remove or rename a category of template (pane 1). This pane lists all the categories of doctrinal templates held in the system. Each category of template refers to a different phase of war (e.g. attack, defence, etc);

b. add or remove a type of template associated with the selected category of template (pane 2). The type of template refers to the type of unit or formation that template represents (armoured, infantry, artillery, etc). This pane displays all the template that were put in for the selected category of templates;

c. add, remove or insert an element for a selected type of template (pane 3). An element refers to a major unit or sub-unit of the unit or formation represented by the selected template. Hence, this pane shows all the elements that should be part of the given template;

d. enter the depth and frontage of a selected type of template (panes 4 and 5). This allows the analyst to adapt the template to the terrain since terrain is a major factor in dictating how forces are deployed; and

e. enter a value for the easting and northing of each element of the selected type of template (panes 6 and 7). This lets the analyst position each element within the template.

This browser will be seldom used during operations. The different doctrinal templates should be known to the EW analyst prior to the start of hostilities and entered in the system at that time. This browser will be mainly used to modify the templates to take account of the variations in the terrain and to changes in doctrine.

As previously mentioned, information is being gathered a little at a time and the EW analyst needs a tactical database to keep track of it all. In this system, he will be able to store all the information gathered from various sources by using the Tactical Database Browser (Figure 6).

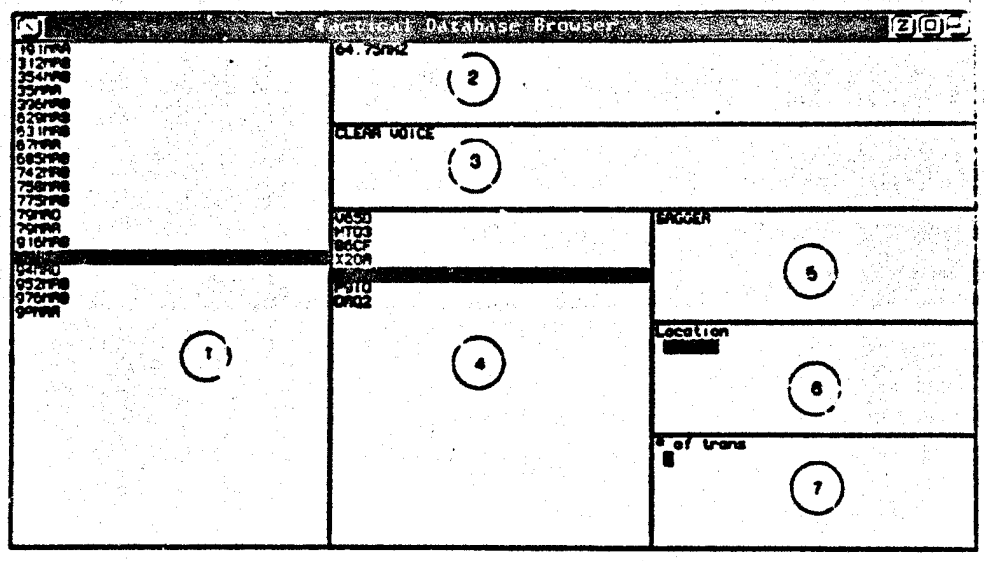


Figure 6 - Tactical Database Browser

This browser also has seven (7) panes and provides the EW analyst with the following information:

a. Pane 1. This pane gives a list of all enemy units and formations held in the tactical database. The EW analyst can add, remove, rename or try to match a unit from this pane. When a unit/formation is selected, the system will display the electronic characteristics of that unit in the other panes of the browser;

b. Pane 2. This pane shows a list of all the frequencies known to be used by the selected unit. The system allows the EW analyst to add or remove a frequency from this list;

c. Pane 3. A list of all types of signal known to be used by the selected unit is displayed in this pane. The EW analyst can add or remove an entry from this list;

d. Pane 4. The pane displays all the nodes (sub-units/units) associated with the selected unit. The system lets the EW analyst add, remove or try to match a node from this pane. Once a node is selected, the system will display the characteristics of that node in panes 5, 6 and 7;

e. Pane 5. This pane displays a list of the different codewords known to be used by the selected unit/formation. The EW analyst can add or remove a codeword from this pane;

f. Pane 6. This pane displays the last known location of the selected node. This information would normally be obtained from direction finding sensors and automatically entered in the database. The EW analyst has the ability to change this information when required; and

g. Pane 7. The number of transmissions emanating from the node is entered in this pane. Again, this number will be updated automatically from information received from the intercept operators positions. The EW analyst also has the ability to change the information in this pane if necessary.

The EW analyst will usually use this browser to obtain the information he needs to provide steerage to the different sensors deployed by the unit. He will also use the information contained in this browser to initiate a search of the inferred (doctrinal) templates to find a match between the deployment of the selected unit/formation (pane 1) or a selected node (pane 4) and a doctrinal template.

As mentioned earlier, the terrain plays an important part in the way a unit/formation commander will deploy his forces. In many instances, the deployment will not conform to the doctrine. Therefore, there is a need to modify the doctrinal templates to conform to the terrain on which the battle is being fought. Also, the terrain will dictate the possible avenues of approach to the friendly position. The EW analyst can change the size and orientation of an inferred (doctrinal) template by using the Inference Calculator. An example of this browser is showed in Figure 7. The calculator has seven (7) panes which allow the EW analyst to :

a. select the desired category and type of template to be inferred. Panes 1 and 2 list all the templates that can be inferred by category (pane 1) and type (pane 2). The lists are identical to the lists found in the Template Database Browser;

b. enter certain attributes for a given template. These are:

1) inference coordinates. Pane 3 lets the EW analyst enter the point on the map where the template will be inferred. It is the intersection of the line representing the FEBA and the line representing the left boundary of the unit or formation for which the template is inferred;

2) angle. Pane 4 allows the EW analyst to enter the angle of the axis of advance (suspected or known) of the enemy forces with respect to the deployment of friendly forces;

3) depth. Pane 5 lets the EW analyst determine the depth for the template that will best match the terrain over which the battle is taking place; and

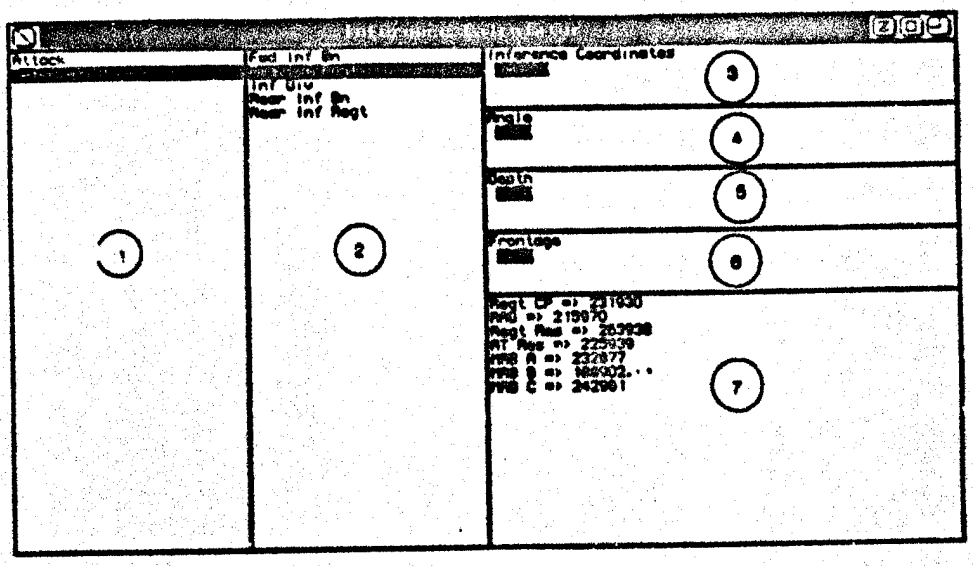


Figure 7 - Inference Calculator

4) frontage. Pane 6 lets the EW analyst determine the frontage for the template that will also best suit the terrain; and

c. add the inferred template to a database of inferred templates. Pane 7 lists all the elements of the selected template with their corresponding inferred location. Each inferred template will be stored in the database of inferred templates with its attributes and a list of elements with their associated inferred location.

The EW analyst will use the Inference Calculator to infer all the templates he feels are appropriate for the type of terrain and the state of hostilities. Since the supported commander is only interested in enemy actions in a specific area of the battlefield, the EW analyst will normally infer all the possible templates that could cover the said area.

After the EW analyst has inferred a set of templates and stored them, he must be able to work with these templates. This can be done by

using the Inferred Templates Database Browser (Figure 8). This browser also has seven panes allowing the EW analyst to get:

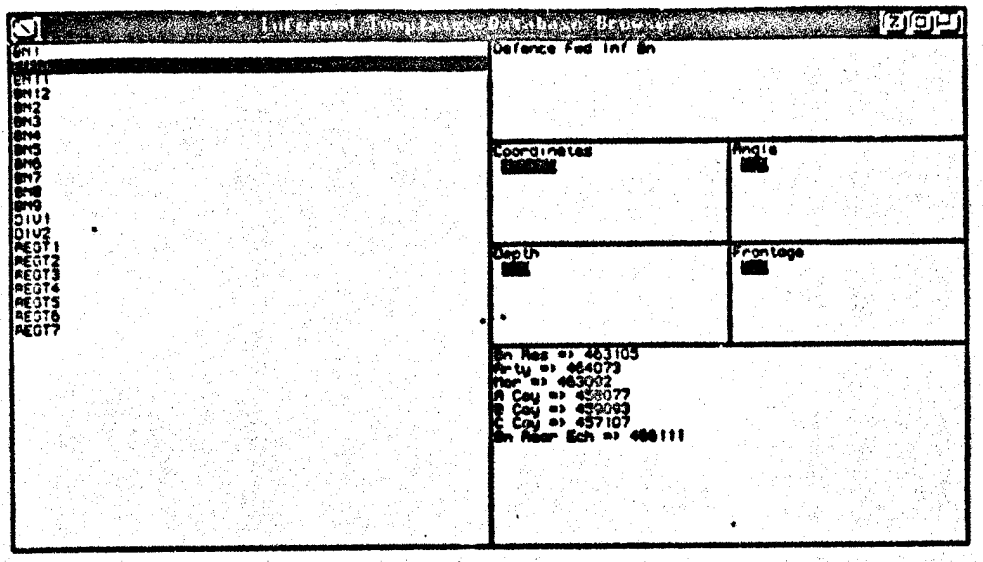


Figure 8 - Inferred Templates Database Browser

a. the list of inferred templates held in the database of inferred templates. Pane 1 allows the EW analyst to remove or rename an inferred template or update the browser;

b. the attributes of the inferred template that is selected. These are:
   1) type of template (pane 2);
   2) inference coordinates (pane 3);
   3) angle of the axis of advance (pane 4);
   4) depth of the inferred template (pane 5); and
   5) frontage of the inferred template (pane 6);

c. the list of all elements associated with the selected inferred template and their inferred location. This pane allows the EW analyst to use the system to match an element of the inferred template to a node in the tactical database or to match the selected inferred template to a unit/formation in the tactical database.

Now, the Ew analyst can use the system to try to deduce the enemy's deployment and intentions as well as to provide steerage to the forward sensors. This can be done in two ways. First, the EW analyst can use the system to match an inferred template he feels best describes the unit/formation he thinks faces our own forces to a unit/formation held in the tactical database. This is done from the Inferred Templates Database Browser. The result of the search will be shown in the Matched Tactical Database Browser (Figure 9).

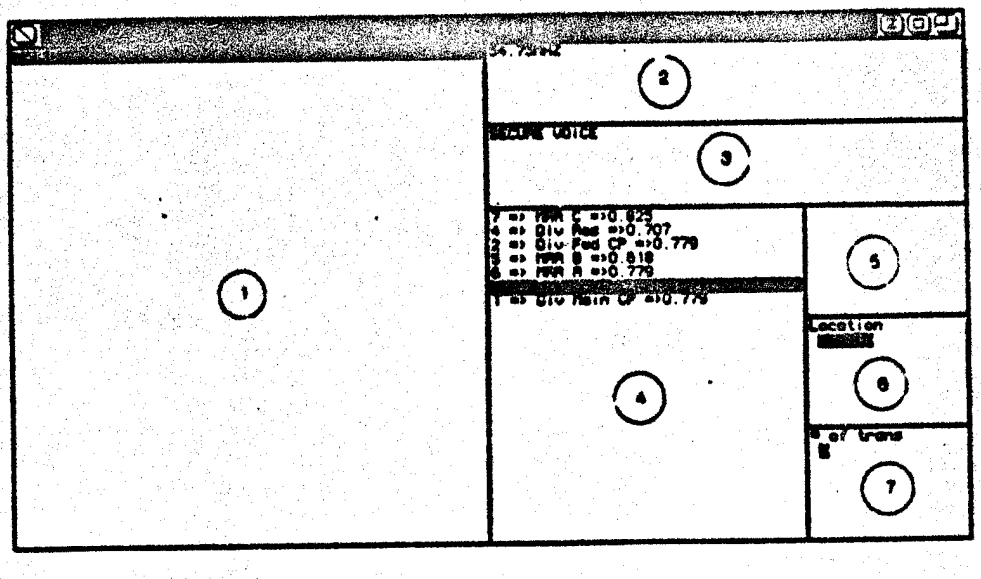


Figure 9 - Matched Tactical Database Browser

This browser is identical to the Tactical Database Browser except for the following:

a. pane 1 lists only the units/formations that have matched the inferred template along with a certainty factor associated with each match; and

b. pane 4 now shows the certainty factor for each element to node match as well as the list of all nodes associated with the selected matched unit/formation.

On the other hand, if the EW analyst knows the identity of the enemy unit/formation facing our own forces but requires more information about it, he can do one of two operations:

a. select the unit/formation from the list in pane 1 of the Tactical Database Browser to gain access to all the information available about the unit/formation that is held in the system. The EW analyst can then use the information to steer the different sensors deployed by the EW unit; and

b. initiate a search of the Inferred Templates Database in order to determine if the selected unit/formation has a match amongst the Inferred Templates. The result of this search will be shown in the Matched Inferred Templates Database Browser (Figure 10).

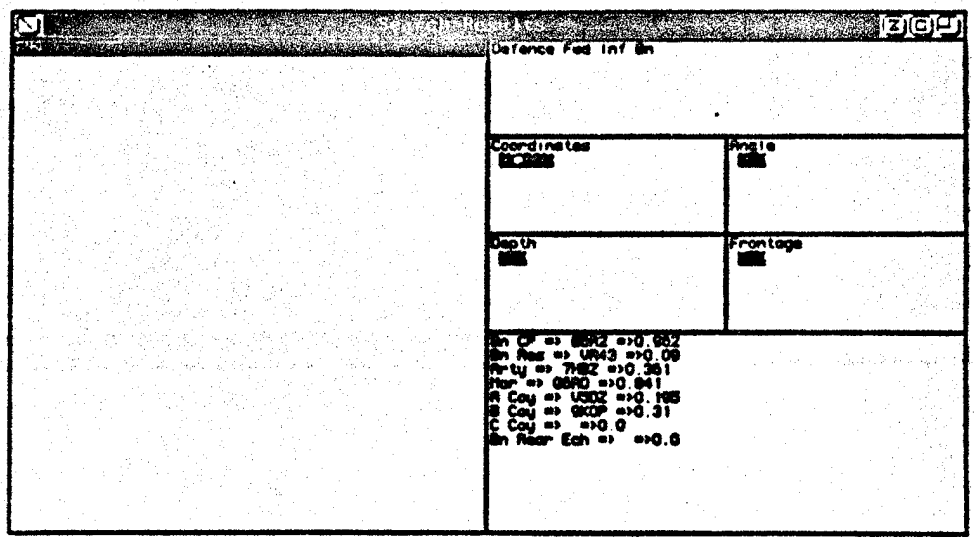


Figure 10 - Matched Inferred Templates Database Browser

This browser is identical to the Inferred Templates Database Browser with the following exceptions:

a. pane 1 contains a list of the inferred templates that have matched the selected unit/formation or a node with an associated certainty factor; and

b. pane 7 includes a list of all the elements of an inferred template with the associated matched nodes including the certainty factor associated with each match.

From this browser, the EW analyst can determine the enemy's intentions from the type of template that matches its actual deployment. He can also determine which information is missing from the comparative list of matches between elements and nodes.

Elements that have no matched nodes indicate that a major sub-unit of the unit or unit of the formation has not been located yet. The EW analyst can then use this information to steer ESM detachments to try to locate the missing sub-units/units to complete the tactical picture.

THE CERTAINTY FACTOR

Many elements will introduce a measure of uncertainty in the attempt to identify the enemy's deployment, strengths and intentions. Consequently, it is important to have a measure of the goodness of any match achieved by the system. These matches are determined using only the location of the elements or nodes. Since the location of an element or node is defined in terms of a value on the "X" axis and a value on the "Y" axis, it is considered a two-dimensional random variable.

The certainty factor (CF) for a match by the system is defined as:

CF = Probability that a point (X_1,Y_1) is within a distance "R" of the point (X_2,Y_2) where

$$R=\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$$

It is assumed the random variable follows a two-dimensional Normal Distribution defined over the area "R" where "R" is defined as $(\Delta X)^2+(\Delta Y)^2 \geq R$. The probability of matching a node to an element is given by

$$P_i = e^{-\frac{R^2}{2}}$$

The CF for matching a unit to a template or vice versa is given by a Chi-Squared distribution with "n" degrees of freedom where n is the number of nodes or elements to be matched [8,9]. The probability of getting a match is given by

$$P = \Sigma_{k=0}^{n-1} \frac{t^k e^{-t}}{k!}$$

where

$$t = \Sigma_1^n -(\ln p_i)$$

CONCLUSIONS

The EWAS represents a giant step forward from the present manual system in use in an EW unit. It provides the EW analysts with the following capabilities:

a. quick access to the technical data held in the tactical database;

b. ability to steer EW and other sensors more efficiently through a quicker access to the technical data required by these sensors;

c. rapid inference of any number of templates that can cover the entire range of deployment possibilities that could be used by the enemy;

d. matching of data from the tactical database to inferred templates to determine the enemy's intentions; and

e. matching inferred templates to actual tactical data to confirm enemy deployments and identify missing units/ sub-units.

These capabilities allow the EW analysts the ability to determine enemy intentions and possible deployments more rapidly and more accurately. This, in turn, allows them to produce more accurate reports to meet the supported commander's PIRs and IRs. EWAS allows EW analysts to cope more efficiently with a large amount of data available from all sources. A demonstration to personnel in the EW units indicated that EWAS is a first step towards the development of a comprehensive system in support of the EW analysts.

RECOMMENDATIONS

The following improvements should be considered to further enhance EWAS:

a. develop or acquire a digital map display system to allow EWAS to display the information on a map;

b. develop a graphics package using icons to allow the display of the templates;

c. develop a software package that will give EWAS the "intelligence" to interface with the mapping system and modify the templates by taking into account the terrain features such as lakes, mountains and built-up areas; and

d. modify EWAS to integrate the tactical database and the database of inferred templates into a single database from which all the information will be obtained.

REFERENCES

1. CFP 321(4), "Signals in Battle - Volume 4 - Tactical Electronic Warfare", Mobile Command Headquarters, St-Hubert, P.Q., 1986.

2. Canadian National Defense, "Fantasian Ground Forces Organizational Guide", Mobile Command Headquarters, St-Hubert, P.Q., 1986.

3. Cox, B.J., "Object Oriented Programming - An Evolutionary Approach", Addison-Wesley, Reading, MA.

4. Digitalk Inc, "Smalltalk/VPM", Digitalk Inc, Los Angeles, CA, 1989.

5. Digitalk Inc, "Smalltalk/V286", Digitalk Inc, Los Angeles, CA, 1988.

6. FM 100 - 2 - 1, "The Soviet Army - Operations and Tactics", Headquarters Department of the Army, Washington,D.C., 1984.

7. GoldBerg, A. and Robson, D., "Smalltalk-80 - The Language and its Applications", Addison-Wesley, Reading, MA, 1983.

8. Kapur, K.C., Lamberson, L.R., "Reliability in Engineering Design", John Wiley, New York, NY, 1977.

9. Meyer, P.L., "Introductionary Probability and Statistical Applications", Addison-Wesley, Reading, MA, 1970.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

SCHEDULING FIGHTER MISSIONS USING AN EXPERT SYSTEM WITH AN OPERATIONS RESEARCH MODULE

Capt J.Genest[1], Capt L.Lefebvre[1], Dr. G.Savard[1], Capt M.Vachon[2]

Abstract

Aircraft serviceability is the ratio between the number of aircrafts ready for flight and the total number available. One of the main goal in scheduling missions is to maintain a high serviceability rate while meeting operational and training requirements. This is a complex problem for the aircraft maintainers since aircrafts may break down unexpectedly after missions making it difficult to follow a preset schedule. We propose in this paper a three-step approach which combines an expert system, an optimization module and a critique module to assist the maintainers and operators in the production of a mission schedule. Our approach first prepares an initial schedule using an optimization model. The expert system then reviews the schedule to enforce domain specific heuristics based on maintainers and operators experience. The schedule is then evaluated by maintainers in conjunction with a critique module which, using machine learning techniques, will refine the heuristics used by the expert system. Our approach is currently being developped for a F -18 squadron using an expert system shell called ART-IM and an operations research system running on a PC. Favorable results were obtained from each module and they are now being integrated in an application running in the Microsoft Windows environment.

[1]Dept of Mathematics, [2]Dept of Administration, Collège Militaire Royal de Saint-Jean, Richelain, Québec

Introduction

Over the past few years aircraft have increasingly employed sophisticated technology designed to reduce pilot workload and improve effectivness. The CF-18's heads-up display and multiple target acquisition systems are examples of such technology. Aircrew in particular have benefitted from the introduction of computers on board. Besides freeing the pilot from tedious details in system operations, computers also provide significant decision analysis capability in target detection, identification and engagement procedures. They make it possible for the pilot to concentrate on flying the aircraft and to better manage the aircraft under various scenarios.

While computer technology is employed in aircraft systems as decision aids for aircrew, it is not used extensively in aviation squadrons for maintenance managers. One area that could substantially profit from automation is that of aircraft scheduling. Serviceability of a combat squadron is the ratio between the number of aircrafts ready for flight and the total number available. This value fluctuates during the day as some aircraft break down after mission completion and others are released by maintenance teams. Aircraft serviceability is critical to operationnal readiness and yet, at present, there is no automated decision support aid available for those who manage operations and maintenance at the squadron level.

Flight scheduling is central to serviceability. When deciding which missions will be flown and when, managers must carefully balance operational, training, and maintenance requirements. Overflying available aircraft will reduce serviceability as will flying certain high fatigue missions. Presently managers base their planning on charts (i.e. tail number vs time to inspection) as well as personal experience and knowledge. The solutions implemented are not always optimal given the conditions and although operational requirements are often met, they are acheived at the cost of undue, painstaking effort.

We are currently developping a Decision Support System/Expert System (DSS/ES) to aid operations managers at the base and the squadron level in optimizing aircraft serviceability. The DSS/ES will combine a mathematical model of the optimization theory with an expert system using domain knowledge about management. It will give the user an insight into the consequences of implementing a given schedule. Machine learning techniques will also be used to make the system adaptive to its environment.

References [1-5] are examples of systems that are intended to assist managers in scheduling tasks. Our approach is the first one to combine concepts from Operations Research and Expert Systems which may solve many problems encountered so far with those approaches.

This paper is organized as follows: first, we will introduce Descision Support Systems concepts. Next, we will show the architecture of DSS F18. Then, we will present the three principal modules of our system and complete with an example based on real-word data.

Decision Support Systems

A decision support system (DSS) couples the intellectual resources of individuals, in our case the operations and maintenance managers, with the capabilities of the computer to improve the quality of decisions. The four major characteristics of DSS are the following:

DSS incorporate both data and models. The decision making process relies on relevant data to be available to guide the manager in the selection of the best alternative. The models are tools designed to evaluate and manipulate the data in order to create and test hypotheses. For example, a model can be a linear program, a statistical method, a scientific computation or an expert system. The DSS will normally provide a user-friendly front end through which the user will call the appropriate model when required. This front end contains a controller program to manage the exchange of the data and the synchronization between the models. Figure 1 shows the architecture of a DSS.

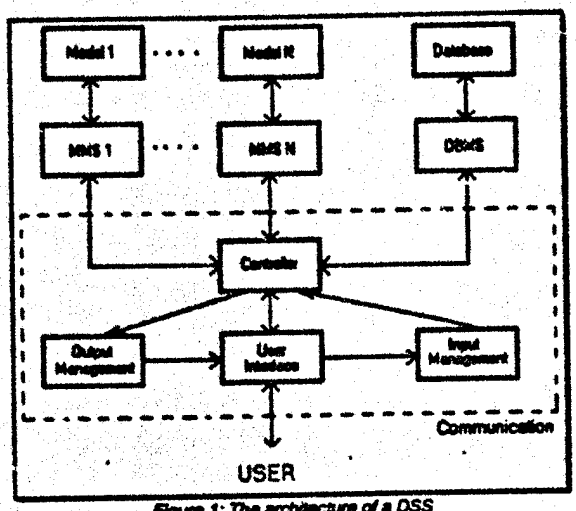


Figure 1: The architecture of a DSS

DSS are designed to assist managers in their decision processes in unstructured tasks. *Structured tasks* always involve the same sequence of steps in its fulfillment and allows no deviations. For example, processing work orders or tracking aircraft flying hours are structured tasks. Structured tasks are best handled using Transaction Processing Systems (TPS) or Management Information Systems (MIS). On the other hand, *unstructured tasks* do not follow a fixed sequence of steps. Examples of such tasks are planning, scheduling and forecasting. The number and order of steps they involved are left to the discretion of the user.

DSS support, rather than replace, managerial judgement. Unlike other types of systems, the user is driving a DSS and not the other way around. The dialogue between the system and the user must always be lead by the latter. This criteria is crucial for the acceptance of a DSS by managers to which the system is targetted.

The objective of DSS is to improve the effectiveness of the decisions, not the efficiency with which decisions are being made. The quality of the decisions made, and not necessarily the speed at which it is obtained, is the ultimate goal of using a DSS. The quality of a decision will be improved if it is based on a large number of data and if a large number of alternatives are evaluated. DSS provide both the access to the data and the models which makes that possible.

The Architecture of DSS F18

The architecture of DSS F18 shown in figure 2 is composed of six models grouped in three categories:

The Mathematical models. This category contains the scheduling and the assignment model. Those two models are based on Operations Research and they are designed to compute the best alternative according to mathematical (quantifiable) constraints.

The Expert/Critique models. The first model, called the scheduling expert model, reviews schedules in order to enforce non-quantifiable constraints. This expert system uses a knowledge-base which includes rules used by experienced operations and maintenance officers. The second model, called the critique model, monitors the scheduling process in order to refine the knowledge-base based on user's preferences.

The Statistical/Data Management System models. The Data Management System, expected to be in production in late 1992, records maintenance data on every location where the F18 is in use. The maintenance data will be used by the statistical model to compute values required by the scheduling and the assignment models such as the probability of aircraft breakdown and flying hours information.

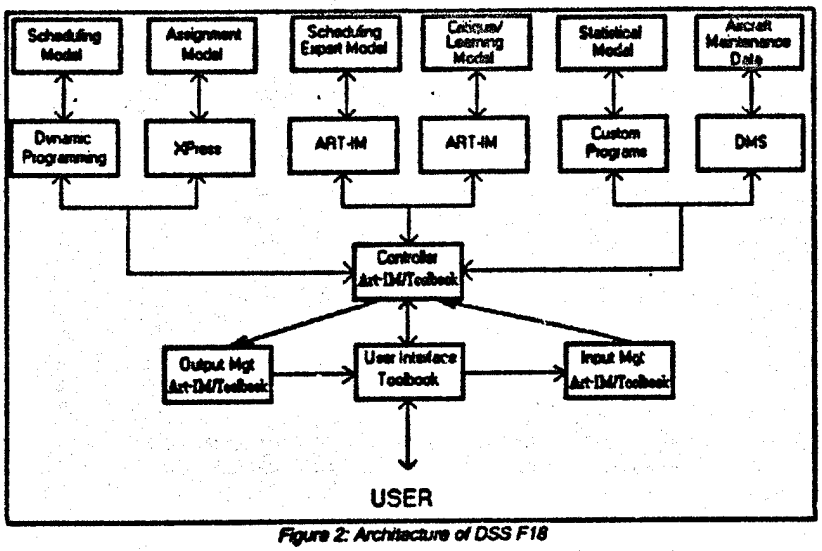


Figure 2: Architecture of DSS F18

The system is managed by a controller program which uses regular object-oriented code and a rule based expert system. The user interface uses state-of-the-art graphics and mouse-driven dialogs to ensure user acceptance and cooperation. The input and output management modules ensures that the data is presented in familiar forms to the user (e.g. calendar pages for scheduling, color graphics for stagger charts, aircraft picture for exercise configuration, etc.)

The Mathematical Model

The mathematical module contains the operations research processes that will conduct stochastic scheduling and assignment of CF-18 aircraft. The scheduling subcomponent receives user input in the form of operational constraints and produces an initial schedule. Based on this schedule as well as maintenance requirements, the assignment submodule will pair each mission with selected aircraft. The initial schedule and the assignment list will then be routed to the expert system which, if need be, will modify these using heuristics developed from the operators and maintainers' experience.

The stochastic scheduling problem can be defined as the problem of finding a set of feasible flight patterns in order to maximize the number of serviceable aircraft and the total number of missions effected during a given time frame. While pursuing this objective certain user defined operational constraints must be met. For example, given missions must be performed on predetermined dates; a minimum number of serviceable aircraft are required to fly certain patterns; a minimum number of sorties must be flown in a specified period, etc. For its part, the aircraft assignment problem is concerned with optimizing maintenance operations. It therefore involves minimizing maintenance human resources while maintaining full flight coverage as well as meeting the different overhaul and maintenance requirements for each individual CF-18.

Work on the mathematical module has focused to date on the scheduling problem. We have developped two models to address the scheduling problem: a deterministic one and a stochastic one. In the first model, the daily expected damage to aircraft is viewed as being soley dependant on the chosen flight pattern. In the second model, the rate of repair is a stochastic process related to the number of unserviceable aircraft and the pattern used.

A deterministic model

Instead of considering the complete distribution of damage to aircraft according to the chosen flight pattern, we consider the average value. A flight pattern is a fixed plan of missions to be flown in a day. The model is thus deterministic and it was formulated as a mixed integer programming model [6]:

$$\max \sum_i \mathbf{nav}_i + \lambda \sum_i \mathbf{tmis}_i \qquad (1)$$

subject to:

$$accp_j\, \mathbf{ptrn}_{ij} - \mathbf{nav}_i \le 0 \quad \forall i \ \forall j \qquad (2)$$

$$\sum_j \mathbf{ptrn}_{ij} = 1 \quad \forall i \qquad (3)$$

$$\mathbf{nav}_i - \mathbf{nav}_{i-1} + \sum_j dmge_j\, \mathbf{ptrn}_{ij} - \mathbf{rep}_{i-1} = 0 \quad \forall i \qquad (4)$$

$$\mathbf{rep}_i - df\{\mathbf{nav}_i - \sum_j dmge_j\, \mathbf{ptrn}_{ij}\} \le df * tac + 0{,}5 \quad \forall i \qquad (5)$$

$$\sum_j tmp_j\, \mathbf{ptrn}_{ij} - \mathbf{tmis}_i \ge 0 \quad \forall i \qquad (6)$$

$$\sum_i \mathbf{ptrn}_{ij} \ge req_k \quad \forall k \in K \qquad (7)$$

$$\mathbf{rep}_i \le urep \quad \forall i \qquad (8)$$

$$\mathbf{nav}_1 = ini \qquad (9)$$

$$\mathbf{ptrn}_{ij} \in \{0,1\} \quad \forall i \ \forall j \qquad (10)$$

$$\mathbf{nav}_i, \mathbf{rep}_i \in \{0,1,2,\ldots\} \quad \forall i \qquad (11)$$

where the constants are:

i : is a subscript that indicates the day, $i = 1, \ldots, m$ where m is the end of the horizon.

j: is a subscript that represents the pattern. $j = 1, \ldots, n$.

k: is a subscript that indicates the ranking of the operational requirement within the set of user defined operational constraints K.

λ: indicates the relative importance of maximizing missions with respect to serviceable aircraft.

req_j : is the j th operationnal requirement specified in the model. An operationnal requirement is the number of times a pattern must be flown.

tmp_j: is the total number of missions flown for a pattern j.

$accp_j$: represents the number of aircraft required to fly a pattern j. Thus this is the maximum number of aircraft flown in a pattern.

$dmge_{ij}$: is the expected number of aircraft damaged after flying a pattern j in day i.

df : indicates the percentage of aircraft under repair that will be returned to service the following day.

tac : is the total number of aircraft (serviceable and non-serviceable) at the squadron.

$urep$: fixes the upper bound of the number of aircraft repaired on a certain day i.

ini : is the initial number of serviceable aircraft on day 1.

and the variables:

$\mathbf{nav_i}$: indicates the number of CF-18 available on a given day i. The origin is day 1 (nav_1) at which the number of available aircraft is known.

$\mathbf{tmis_i}$: represents the number of missions flown on a given day i.

$\mathbf{ptrn_{ij}}$: stands for pattern j on day i. It takes on a value of 0 or 1: 0 when the pattern is not used on that day and 1 when it is.

$\mathbf{rep_i}$: is the number of aircraft repaired on day i.

The equations in the model may be interpreted as follows:

Equation (1) represents the objective i.e. to maximize the number of available aircraft and the number of total missions for each day within the horizon;

Equation (2) indicates that in order to to fly a daily mission pattern on a given day, the number of available aircraft on that day must be superior or equal to the number of aircraft required for

that pattern;

Equation (3) states that only one pattern will be used on each day;

Equation (4) states that the number of available aircraft on a given day is equal to the number of serviceable aircraft on the previous day minus the expected number of aircraft damaged during flight plus those repaired in the meantime;

Equation (5) defines the number of aircraft repaired during a given day as a percentage of the aircraft that are unserviceable at the beginning of that day and the aircraft that are damaged during the day. The addition of 0.5 to the resulting number serves to approximate the number of aircraft repaired to the next higher number;

Equation (6) states that total number of missions flown on a given day is equal to the number of missions for the pattern flown;

Equation (7) specifies that a given mission must be flown at least a user defined number of times;

Equation (8) defines an upper bound on the number of aircraft that can be repaired on a given day; and,

Equation (9) gives the number of aircraft available at the origin.

This model can be solved by a branch and bound method, but the computation time necessary to obtain optimal results on a realistic problem is unacceptable for an interactive system. Hence, we have exploited the particular structure of this model and solved it by a decomposition approach. If the program is relaxed by omitting the operational constraints (8), the resulting model can be formulated as a shortest path problem. Based on this, an algorithm using a dynamic programming approach [7] was developped to solve the relaxed problem in real time. The recursive function used is

$$v_i = \max_j \{v_{i+1}(j) + \mathrm{nav}_i + \lambda \mathrm{tmis}_i(j)\}$$

where: v_{i+1} represents a feasible state at stage $i+1$ given the pattern j chosen at stage i.

The operational constraints are taken into account by an implicit enumeration over the feasible set defined by these constraints. Under the terms of the preceding deterministic model the objective was unambiguously specified by fixing values for the decision variables. Preliminary results give us interesting insights on the behaviour of the system.

A stochastic model

Statistical analysis conducted parallely on CF-18 flight data revealed that the rate of repair is more likely a stochastic process related to the number of unserviceable aircraft, and the pattern used. For simplicity, let us define:

$x^i = (nav_i, ptrn_i, rep_i, tmis_i)$

and

ξ^i: is the vector of the probability that given a number of serviceable aircraft for the day and a pattern there will be a certain number of mission-capable aircraft on the next day. A discrete distribution with $m(j)$ values for pattern j is assumed.

Then the model can be written as;

$$\max_{x^1 \in S^1} E\{nav_1 + \lambda tmis_1 + Q_1(x^1, \xi^1)\} \tag{12}$$

$$Q_{i-1}(x^{i-1}, \xi^{i-1}) = \max_{x^i \in S^i} \{nav_i + \lambda tmis_i + E[Q_i(x^i, \xi^i) : \xi^1, \ldots, \xi^{i-1}] \quad s.t. \quad T^{i-1}x^{i-1} + W^i x^i = \xi^{i-1}\} \tag{13}$$

with $Q_i \equiv 0$ for $i = n+1, n+2, \ldots$.

where S^i corresponds to constraints (2 - 3), (6) and (8 - 11) for a given i and T^{i-1} and W^i are the linking matrices corresponding to the number of aircraft available after the random variable ξ^i is observed. These constraints replaced the constraints (4 - 5). This model is an n-stage stochastic program [8] with fixed recourse. However, adding the operational constraints (7) breaks the block angular structure of this program.

As for the previous model, we have exploited the structure of this model and solved it by a decomposition approach. The subprogram is still a shortest path problem (the number of nodes and arcs increased but remains tractable). Again we solve the relaxed program by dynamic programming and the recursive function used is now:

$$v_i = \max\left\{\sum_{m(j)} \xi^i_{m(j)} (v_{i+1}(\xi^i_{m(j)}) + \text{nav}_i + \lambda \text{tmis}_i(j))\right\}$$

Again the operational constraints are taken into account by an implicit enumeration over the feasible set. Research in the mathematical module will now shift towards the development of methods to solve the scheduling problem with more general operational constraints and to resolve the aircraft assignment problem.

The Statistical Model

The mathematical model is based on the assumption of daily patterns in the flying and repair operations. These patterns are identified by a statistical analysis of the records in the data

management system (DMS). It is a continuous process to capture the pattern changes arising from modifications to such aspects as squadron role, fleet strength or training of groundcrew and aircrew.

The factors creating a pattern are:

1. Total number of aircraft available at the start of a day.
2. Total number of missions flown in a day.
3. The mission types flown in a day.

A probability distribution is created by pattern for the number of aircraft unserviceable on the next day. The link with DMS allows a continuous upkeeping of the probability values. The daily user's raw inputs are filtered by the statistical model to keep only the information of interest for probability estimations. If an updated probability exceeds by a set margin the previous value, the probability distribution mathematical expression is reestimated for all probabilities of this pattern and the dynamic program matrix of data corrected accordingly.

The present model is made of 20 patterns occurring 5 to 25 times each over the year 1990 at 433 Squadron, CFB Bagotville. The probability distribution for each pattern is estimated based on the least square fitting method. The observed probabilities fitted are the number of occurrences for X aircraft unserviceable on the next day for a given pattern over the total number of occurrences for the fixed values of the three factors making this pattern. The estimated probabilities are sent to the dynamic programming model along with the information about the factors making the pattern. The estimation of the probabilities is presently based on one year of operations and will be extended to two years once the DMS supports such a databank. A seasonality factor will then be introduced to account for block leaves, severe winter weather and intensive exercise periods.

After a significant time of operation of the decision support system, the expert system model will analyze the choices of pattern for optimal solution. The pattern barely used will be highlighted. The users will be informed and asked to insert new patterns if desired. In a positive case, the statistical model is triggered for the new patterns, closing the loop of the system.

The Expert/Critique Model

The purpose of the Expert/Critique Model is to:

1. Consider human factors in the scheduling process,
2. Perform sensitivity analysis on the mathematical model,
3. Monitor the performance of the mathematical model, and,
4. Monitor user's response to proposed alternatives.

Human factors, such as technician workload and morale, are sometimes difficult to reconcile with operational requirements. Unfortunately, human factors are non-quantifiable and cannot be integrated in the mathematical model. Successful managers use their experience to

evaluate the impact of aircraft missions schedules on the people and the organization. They also use their experience to determine which modification can turn a weak schedule from the human factor point of vue into a better one.

Expert systems [9] are the best technology available today to model managers experience. The heuristics (or rules of thumb) they use can be translated into patterns-actions (production) rules where patterns represent schedule portions and actions represent modification to the schedule. For instance, if a rule recognizes that a certain part of the schedule will require a great deal of overtime, this part can be re-adjusted by a rule that reduces overtime while respecting operational requirements.

Sensitivity analysis is the process of adjusting the sensitive parameters (i.e. those that cannot be changed without changing the optimal solution). The purpose can be to identify the best tradeoff between two conflicting goals such as, for instance, aircraft missions and serviceability, or to modify certain results of the model without changing the optimality of the solution. In some cases, the sensitivity analysis will be performed by the expert system on its own. In other cases, it will be performed by the user with the assistance of the expert system.

There is a major benefit in having the expert system assist the user in conducting sensitivity analysis. The benefit is that the user will be able to detect and change some parameters without ever coming into contact with the actual mathematical model. The learning component of the Expert/Critique Module will monitor the sensitivity analysis actions initiated by the user in order to suggest similar actions in future scenarios. The outcome of the sensitivity analysis actions will determine how relevant they are and if they are worth remembering.

The user of DSS F18 is free to impose any constraints (called user-imposed constraints) to the model before scheduling is performed. The expert system will monitor the impact of those user-imposed constraints on the performance of the scheduling model and on the quality of the resulting schedule. The objective is to provide guidance to the user in the choice of these constraints by providing a preview of the impact on the scheduling process and on the schedule. Guidance includes both encouraging and discouraging the choice of a user-imposed constraint.

In several cases, user will be presented more than one alternative for a scenario. The Expert/Critique Model will monitor which alternative is prefered by the user and will evaluate the outcome of the alternative for the scenario using user feedback and data from DMS. When a similar scenario is presented to the user, the Expert/Critique Model will indicate which alternative is the most likely to payoff and which alternative is to be avoided. The expert system will provide justifications for its recommendations and the user will retain the ultimate responsibility for the decision taken.

Example

Testing was conducted on both deterministic and stochastic models using fictitious data resembling real-life situations. To illustrate the DSS-F18 concept, we limit ourselve to an example using the deterministic model. A simple scenario that meets the following criteria is chosen:

- the envisaged horizon was 10 days;

- aircraft effected three series of sorties per day;
- patterns with 12 to 18 sorties per day were retained;
- waves of 2 aircraft were used for interception missions, 4 for air combat manoeuvers,and 6 for tactical missions;
- the maintenance to mission factor was arbitrarily chosen to be three (3); and,
- the maintenance crew can repair aircraft up to a maximum of three aircraft per day.

This scenario represents the scheduling process used by a squadron for an horizon of 10 days. In this horizon, the squadron must participate in three exercises: a ground attack exercise on day 6, an air refueling exercise on day 7, and an air defence exercise on day 8. An additionnal constraint for the squadron is that twelve air defence missions have to be flown at least once outside the exercise period.

Table 1 shows that for this scenario, 15 patterns containing intercept, tactical, or air combat manoeuver missions with 12 to 24 sorties were input into the model. The exercises selected correspond to pattern 8 (ground attack), pattern 15 (air refuling) and pattern 4 (air defence). Note that the special request for pattern 4 and the air defence exercise.

No	*Patterns*			*Total Sorties*	*Expected Damage*	*Repairs*
1	2	4	6	12	1	0.6
2	6	2	4	12	2	1.2
3	4	6	2	12	3	1.8
4	2	6	4	12	4	2.4
5	6	4	2	12	5	3.0
6	4	2	6	12	6	3.6
7	4	4	4	12	3	1.8
8	6	6	6	14	4	2.4
9	4	6	4	14	1	0.6
10	4	4	6	14	2	1.2
11	6	4	4	14	3	1.8
12	6	6	4	16	1	0.6
13	6	4	6	16	2	1.2
14	4	6	6	16	3	1.8
15	8	8	8	24	4	2.4

Table 1: Input data for the deterministic example

Figure 3 presents the dialog screen used to enter the scenario. This screen represents a month calendar where the user uses a mouse to drag icons representing exercises to specific dates. For example, the user inserted the icon for ground attack on the 17th. The user enters fixed constraints by double-clicking on a specific date and the number of missions for the day. When the dates representing the horizon are selected, the user clicks on the button Schedule on the bottom

left of the screen to call the mathematical model which will produce the optimal schedule.

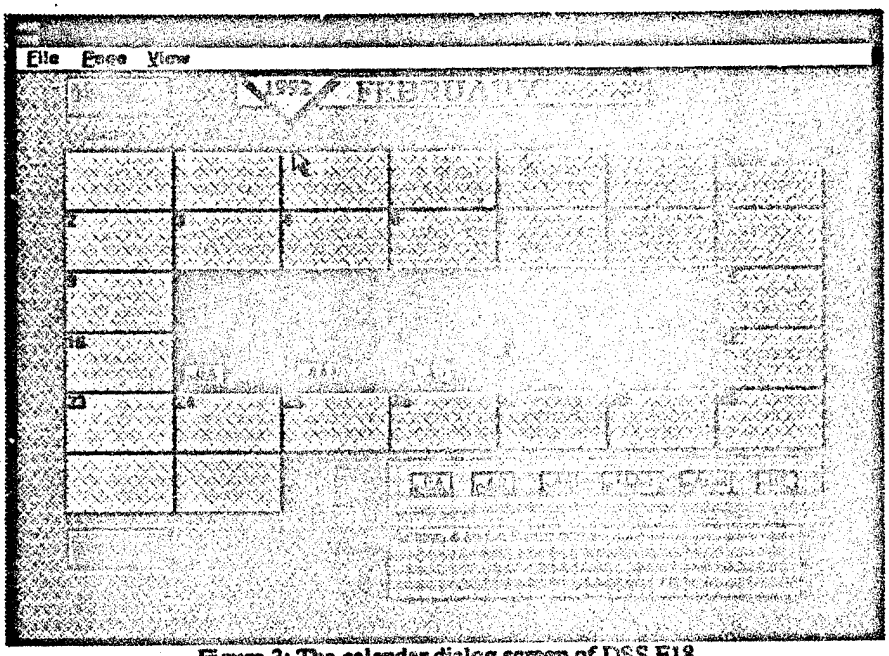


Figure 3: The calendar dialog screen of DSS F18

The results for the optimal schedule are presented in table 2. We observed that in the last days, once the model has satisfied all constraints, it focuses on maximizing missions to the detriment of serviceable aircraft. This entails a decrease in serviceable aircraft over the last few days. This is called *end of period effect* and can be corrected by extending the horizon beyond the user's request or by introducing a salvage value in the objective function.

Day	*No Available Aircraft*	*No of Missions Effected*	*Flight Pattern Used*
1	12	16	12
2	12	12	4
3	10	16	12
4	11	18	8
5	10	16	12
6	11	14	8
7	10	24	15
8	9	12	4
9	8	18	8
10	7	18	8

Table 2: Results for the deterministic scenario

Figure 4 shows the screen displayed by the system after the expert system analysed the mathematical model solution in order to account for qualitative factors. This screen shows the solution and some recommendations for its implementation. The total number of missions to fly

each day is displayed with the initial constraints highlighted.

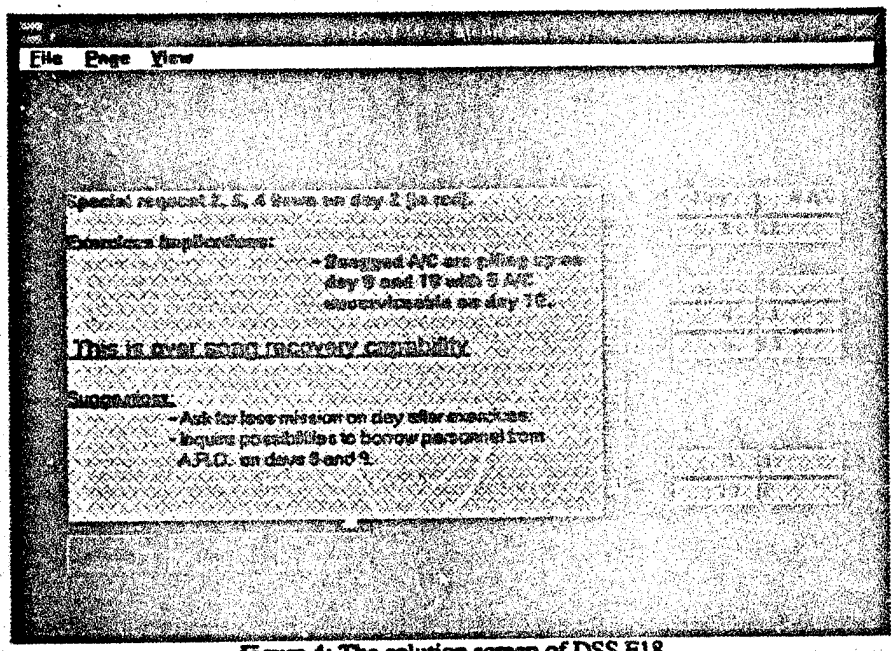


Figure 4: The solution screen of DSS F18

Concluding Remarks and Further Research

Research and development continue on each module and on the full integration of them. For the mathematical module, we are presently working on the generalization of the decomposition approach for taking into account different types of operational constraints. The assignment model is currently under development and has yet to be integrated in the system. We are also considering an approach based on stochastic process with possible perturbed transition matrix. The statistical model will explore the possibility of seasonality and the commonality of patterns probability distribution between F-18 fighter squadrons across Canada. Also the conception of an automatic program for analyzing the upcoming data from the database DMS will be adressed. In the expert/critique module, a major knowledge engineering exercise has to take place to create and validate rules to represent the experience and knowledge of operations and maintenance managers.

Acknowledgment

We would like to thank Major Vos, from Director Fighter Trainer Engineering and Maintenance 2-7 for his support for this project. The comments from Majors Bourret and Burke on our initial prototype helped us target our system to the maintenance manager. Locally, we thank our statistician Nancy Doucet and our programmer Frederic Lesieur.

References

1. Johnson, L.A., and Laase, E.R., "A Policy Model of USAF Aircraft Operational

Reliability", Masters Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, September 1979.

2. Lewellen, M.L., "A Simulation Model for Determining the Effect of Reliability and Maintainability on maintenance Manpower Requirements and Mission Capabilities", Masters Thesis, Air Force Institute of Technology Wright-Patterson AFB, Ohio, December 1985.

3. Hargrove, A., "Three Computer Based Aids to Maintenance Scheduling", School of Engineering Tuskegee Institute, Alabama, September 13, 1982.

4. McCaffrey, M.J., "The Feasibility of Implementing an Expert System for Aircraft Maintenance Discrepancy Scheduling with the Naval Aviation Logistics Command Management Information system (NALCOMIS)", Masters Thesis, Naval Postgraduate School, Monterey, California, September 1985.

5. David, L.A., and McSwain, W.R., "Naval Aviation Maintenance Decision Support System", Masters Thesis, Naval Postgraduate School, Monterey, California, March 1989.

6. Nemhauser, G.L. and Wolsey, L.A., *Integer and Combinatorial Optimization*, Wiley, New York, 1988.

7. Denardo, E.V., *Dynamic Programming Models and Applications*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

8. Prekopa, A. and Wets, J.-B., *Stochastic Programming 84: I & II*, North-Holland, Amsterdam, 1986.

9. Jackson, P., *Introduction to Expert Systems*, Addison Wesley, Menlo Park, 1986.

10. Turban, E., *Decision Support and Expert Systems*, MacMillan, New York, 1988.

11. Hopple, G.W., *The State of the Art in Decision Support Systems*, QED Information Sciences, Welleslay, 1988.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

A FIRST GLANCE AT REAL-TIME KNOWLEGE-BASE SYSTEMS FOR THE THREAT EVALUATION AND WEAPON ASSIGNMENT PROCESS

R.Carling[1]

Abstract

The anti-ship missile which is launched from a submarine, surface ship or air platform can be a serious threat to Canadian shipping. The Threat Evaluation and Weapon Assignment (TEWA) process is a very important function in a warship's Command and Control system since it ranks the air threats which are detected by the automatic detection and tracking process and assigns a hardkill or softkill weapon to these ranked threats.

A knowledge-base TEWA has been built in SMALLTALK/HUMBLE to do Threat Evaluation and Weapon Assignment for a single stationary TRUMPlike ship attacked by anti-ship missiles. The TRUMPlike ship is equipped with SM-2 missiles, a medium calibre gun, a close-in weapon system and suitable softkill weapon systems. This paper will describe the knowledge-base TEWA for the TRUMPlike ship and give results of its performance in multi-threat scenarios. It will also describe current efforts to have the TEWA assign hardkill/softkill weapons to anti-ship missiles equipped with radar, infrared or ARM (anti-radiation missile) seeker heads from a ship which can manoeuvre.

Finally, the author will present a list of real time requirements that the knowledge-base must satisfy when it does threat evaluation and weapon assignment for a single ship in a self-defence role. He will describe the problems to be overcome in building a real-time knowledge-base TEWA for Canadian ships. He will also give qualitative descriptions of research efforts to find solutions to these problems.

[1]Defence Scientist, Defence Research Establishment Valcartier, Quebec City

2

INTRODUCTION

The defence of a frigate or destroyer from an attack of anti-ship missiles launched from submarines, enemy surface ships or air platforms constitutes a very serious problem to the ship designer and to the design of specific systems and software making up its anti-air warfare combat system. The problem is a real-time problem since the threats are travelling at very high speeds thus leaving the ship only a few seconds to do the Threat Evaluation and Weapon Assignment and fire weapons at the threats to destroy them. It became evident that a fully automatic, computerized system was essential to fulfil this real-time requirement. There was also a need to study the methods involved in ranking the air threats at each second of an air attack and a need to study the decision-making process for assigning the most suitable hardkill/softkill weapons to these threats.

An approach using knowledge-base systems has been adopted to study the TEWA process. The air defence functions involved in the process were studied and the reasoning methods from artificial intelligence were associated with possible applications of the air defence problem. The outcome of this knowledge review process was a list of areas in ship air defence where artificial intelligence methods could be applied. The design for the knowledge-base system was based on a five function battle management process. Originally, it was perceived that a six function battle management plan would be sufficient to model the defensive reactions of the ship including those involving weapons and sensors. Since this study only involves reactions involving weapons, the sixth function that selects between reactions involving weapons and those involving only sensors was removed. In order to test and analyze the decisions made by the knowledge-base system, a framework environment called a TEWA simulator was created in an object-oriented programming language. The final arrangement for testing the knowledge-base system consisted of a TEWA track generator which generated air tracks for prescribed threats attacking the ship, and a TEWA simulator which contained objects representing all the important elements of the ship's AAW combat system (e.g. fire-control radar, missile launching system, close-in weapon system, etc) and representing the external objects of the simulation (ship platform, air threat, and missile). The knowledge-base system was tested by running the TEWA track generator first to generate air tracks and then submitting these tracks to the simulator. The knowledge-base system ranked the tracks according to its rules and then produced weapon assignments which caused the state of objects in the simulator to change (e.g. the missile launcher goes from a loaded state to a firing state). The threat rankings, the weapon assignments and the changes of state of objects in the simulator were

recorded in a report generator produced by the TEWA simulator program.

BACKGROUND

The Canadian Forces have a requirement to protect their warships and merchantmen from air threats (aircraft and anti-ship missiles). Their warships should, therefore, be able to defend themselves in a self-defence role and protect convoys of merchantmen in an area defence role. It is essential for the warship to develop a complete picture of the tactical situation around it in preparation to ranking the threats and assigning weapons to them. Artificial intelligence has already been successfully applied to the tactical situation evaluation level of multi-sensor data fusion for a convoy of ships having different functions, see references [1] to [5].

It is known that knowledge-base systems can be successfully applied to four important classes of problems : diagnosis, control, simulation and design, see references [6],[7],[8] and [9]. In fact, one of the first expert systems to be used successfully was MYCIN, an expert system for diagnosing the presence of diseases in a patient and specifying the remedy. Some current diagnostic expert systems use ideas developed in MYCIN. The uncertainty reasoning used in the knowledge-base system described in this article is based on ideas coming from Mycin. Therefore, the work described in this article constitutes an exploratory study to see whether artificial intelligence can be applied to the TEWA function of an AAW frigate or destroyer.

Recently, a real time knowledge-base demonstrator to do Threat Evaluation and Weapon Assignment has been set up. It is known as RRASSL (Reaction Resource Allocation-Single Ship Level), see reference [10], and it automates the TEWA process. It includes the ability to recommend the optimum combination of missiles, guns, jammers and decoys in order to maximize their effectiveness and prevent mutual interference.

The US Navy has software packages using expert systems to perform decision aids, see reference [11]. CASES (Capabilities Assessment, Simulation and Evaluation System) supports the analysis of naval operations comprising combinations of ASW, AAW and strike warfare. CASES also incorporates two expert systems to assist planners in setting up and analyzing different warfare scenarios.

Reference [12] gives an account of a Tactical Situation Evaluation function and Tactics Planner (including weapon assignment) for fighter aircraft. The fighter aircraft must monitor the skies for the presence of hostile aircraft and missiles and must take defensive

action against them. The AAW frigate or destroyer must do the same thing but it is at a disadvantage because it cannot manoeuvre as rapidly as a fighter aircraft.

The purpose of the TEWA knowledge-base system is to assess the feasibility of implementing a ship's TEWA function using Artificial Intelligence (AI) technology. To conserve project funds while focusing on TEWA related issues. the facility has been implemented with the following assumptions and limitations :

(a)Only a single ship's TEWA is considered and no other warships are available;

(b)The ship is not maneuvering;

(c)All threats, sensors and weapons are modelled at the lowest level of fidelity which is consistent with achieving the objectives of this project.

(d)All anti-ship missile threats have active radar homing heads.

Limitation (a) is not considered to be too serious, since an AAW simulator is currently being developed in which air attacks on a convoy of warships will be simulated. The convoy of warships in the simulator will operate under a distributed architecture, i.e., the TEWA of each ship will be autonomous and will be a copy of the knowledge-base system described in this article. Limitation (c) is also not considered to be serious since this project is a TEWA study and the threat, sensor and weapon models have been built with sufficient detail to provide all the necessary inputs to the TEWA and to deal with all the outputs from TEWA. Limitation (b) is more important since in some cases a ship has to clear blind arcs through rotation before being able to use missile and gun systems and in many cases it has to rotate to deploy chaff against an anti-ship missile threat. This deficiency will be corrected in the AAW simulator where ship maneuvers will be simulated before chaff deployment and missile firings when they are required. Limitation (d) is considered to be reasonably serious since there are anti-ship missiles with infrared and anti-radiation missile seeker heads. At the present moment work is ongoing to develop an adequate Target Evaluation function for these threats and to extend the resource planning of this article to these threats.

The TEWA process itself is implemented at the highest level of fidelity using AI techniques, consistent with the availability of funds. In general, the TEWA process can generate three types of reactions in order to deal with a given situation :

(1) Defeat the threat through the use of hardkill and softkill weapons;

(2) Improve force knowledge through sensor cueing or special processing;

(3) Improve force efficiency through suitable force ship formations and task allocations.

As stated in the introduction, the TEWA Track Generating Process is separate from the TEWA Simulator process. The sensor models used to sense the threats are relatively simple. Consequently, the TEWA function is not able to cue sensors to improve force knowledge. Also, because the facility considers only a single warship which cannot manoeuvre, there are no other force resources to be accessed in time of need. Consequently, reactions of type 2 and 3 are not supported in the knowledge-base system and the TEWA design presented will only attempt to generate type 1 reactions. The sensor cueing option is a technique used in electronic countermeasures and becomes very important in studying the behaviour of the Command and Control system in electronic warfare.

THE KNOWLEDGE REVIEW PROCESS

An initial knowledge review was made of the TEWA process to facilitate the choice of knowledge-base shell and the simulation environment in which it would be tested. After doing the study, it was evident that the following artificial intelligence functions were required for the study. These functions are described in the following paragraphs. After examining various artificial intelligence shells, the shell HUMBLE was chosen because it integrates easily into a SMALLTALK environment and was found to be one of the best shells for supplying the functionality required for our study of TEWA involving anti-ship missiles.

In the initial analysis of the TEWA problem, forward and backward chaining were both given emphasis since their use separately and in combination has become standard in knowledge-base applications. Reference [13] points out that backward chaining is often more efficient and rapid than forward chaining because given an objective only those rules are fired which produce intermediate values leading to an evaluation of the objective, whereas in forward chaining rules are often fired to produce intermediate values which are not used to evaluate the objective. This was found to be true in the design of the TEWA knowledge-base.

There is an important technique in knowledge-base systems called non-monotonic reasoning. This is essentially making a guess (default value) in the absence of facts and being able to clean out default-based inferences when a contradictory fact arrives. Non-monotonic reasoning was initially identified as a necessary technique in connection with determining the target of an anti-ship missile. After building the target evaluation function of this knowledge-base system, it was realized that the estimates of kinematic parameters

would indicate that the target was nearing the ship in spite of sensor errors and hence non-monotonic reasoning was not required in this modelling of an attack by anti-ship missiles.

A requirement has also been identified for the compact handling of multiple confidence factors, and multiple confidence factor combination paradigms. Incoming data to TEWA from sensors is usually prone to errors and therefore confidence factors may be assigned to the sensor measurements. In addition, the knowledge-base may assign confidence factors to the inferences that it makes.

Temporal reasoning has a potential application in the threat evaluation process. It involves time intervals and logic calculations applied to them. The threat evaluation process will use temporal reasoning in that present threat ranking values will depend on past ones. Spatial reasoning has a potential application in engageability calculations. When a hostile target is outside the engageability envelope of a weapon system a projection from the current trajectory is required to see when and where it enters the weapon's engageability envelope.

In a blackboard-organized knowledge-base system, several experts co-operate in the problem solving under the supervision of a controlling expert who reasons about the efficiency of the search for solutions. A shared database is the "blackboard", while the control module may have its own local database to keep track of the evolving solution state. For a single ship attacked by anti-ship missiles, the current TEWA design involves five air defence functions which send information to a central TEWA unit. A blackboard structure is therefore a natural choice for this formulation of the problem with the control module being the central TEWA unit and the inputs coming from four HUMBLE knowledge-bases.

The list of tracks to be input to TEWA, as well as various tables of probabilities and sensor errors, weapon characteristics and historical (temporal) representations of tracks all point to the need for very flexible data structures within the knowledge-base system. For this TEWA problem, the SMALLTALK object has been found to be adequate for storing this kind of knowledge.

THE TEWA TRACK GENERATOR

The TEWA Track Generator code creates a track data file which is used as input for the TEWA Simulator. The data within the file represents typical output over time from an automatic track management system of a ship being attacked by anti-ship missiles. The output from the TEWA Track Generator is not generated in real time and it is the author's opinion that the error in track attributes coming from this programme is about two to three

times smaller than the attributes of operational track data. The overall plan for the TEWA Track Generator is given in figure 1. The Operator configures the scenario consisting of anti-ship missiles, ship's sensor suite and environmental conditions by submitting the appropriate data to the SMALLTALK man machine interface. The target tracks generated are for anti-ship missiles attacking a single TRUMPlike ship equipped with a long range radar, a medium range radar, an electronic support measure, an infrared search and track device, the two types of fire-control radar needed to illuminate targets for the SM-2 system (Standard Missile-2) and the fire-control radar for the 76 mm gun. There are four fixed vertical profiles for the anti-ship missile threats : sea skimmer, high diver, shallow diver and hybrid diver. In the interface, the user specifies the number of threats, the vertical profile of each one, the attack direction and he must choose a subset of the sensors mentioned above. The target track generator computes an interpolated track from the user supplied waypoints of each threat. There is a surveillance radar model that simulates long and medium range radar detections, a passive sensor detection model that simulates both IRST and ESM detections, and a fire-control radar model that generates fire-control radar tracks. It is assumed that the fire-control radar must be operating in tracking mode for the ESM to be able to function. If the fire-control radar is illuminating, the ESM must be switched off.

The data from the surveillance radar and passive sensors undergoes data association, data correlation and finally sensor fusion. The latter is performed by using a minimum variance technique to give prefused surveillance tracks, i.e., for a given range the fused data is the sensor track data with the smallest variance. If the fire-control radar is illuminating or tracking (ESM is switched off or on respectively), its tracks are fused with the surveillance tracks in a final fusion stage.

The Higher Order Attributes function in figure 1 uses information such as position, velocity, acceleration, and ESM mode to assign an identification to the track, (i.e., plane, anti-ship missile or even a more specific identification such as the type of anti-ship missile e.g. Exocet). At the present moment, none of the methods of handling uncertainty mentioned in reference [4] are used in track identification. The identification of each threat is supplied at each data record where ESM data is available. A confidence factor is associated with the identification which is calculated as a function of the distance between the threat and the ship.

8

THE TEWA SIMULATOR

The basic design of the knowledge-base TEWA is given in figure 2. The goal of the TEWA function is to assess the prevailing situation and make a command decision regarding the use of hardkill and softkill weapons against the perceived threats. It consists of the major subfunctions : Target Evaluation; Result Evaluation; Force Resources Evaluation; Candidate Reaction Evaluation; and Rank Candidate Reactions.

THE TARGET EVALUATION FUNCTION

The main function of the Target Evaluation function is to assess the situation in the airspace around the ship and to generate a set of propositions about the established target tracks. These propositions will be in the form of a threat list containing a determination of the level of threat posed by each track. Each of these propositions will be characterized by a confidence factor which indicates the strength of belief in the given proposition. In later stages of the TEWA process, this confidence factor will play a role in determining reaction ranking.

The Target Evaluation function is triggered by the arrival of track data from the Read Data Frame function. This is accompanied by a Frame Ready signal. If no Reset signal has been received, then the Target Evaluation function will over time, build up a perception of the situation around the ship, and update it constantly as new data arrives.

As described above, the Target Evaluation function generates threat lists and distributes them to the Candidate Reaction Evaluation function, and to the Rank Candidate Reaction function. This is accompanied by a Threat List Ready signal.

The purpose of the Target Evaluation knowledge-base is to determine whether an arbitrary track is a threat and, if it is considered to be a threat, quantify the level of threat. A set of rules are provided in the knowledge-base. Each track is "submitted" to each rule. Each rule that is satisfied contributes evidence to one set of limited propositions concerning that track.

Each rule is associated with a certainty factor. The certainty factor associated with each rule is a measure of the reliability of that rule. Rules are written in the following format :

If A then X with certainty factor CF

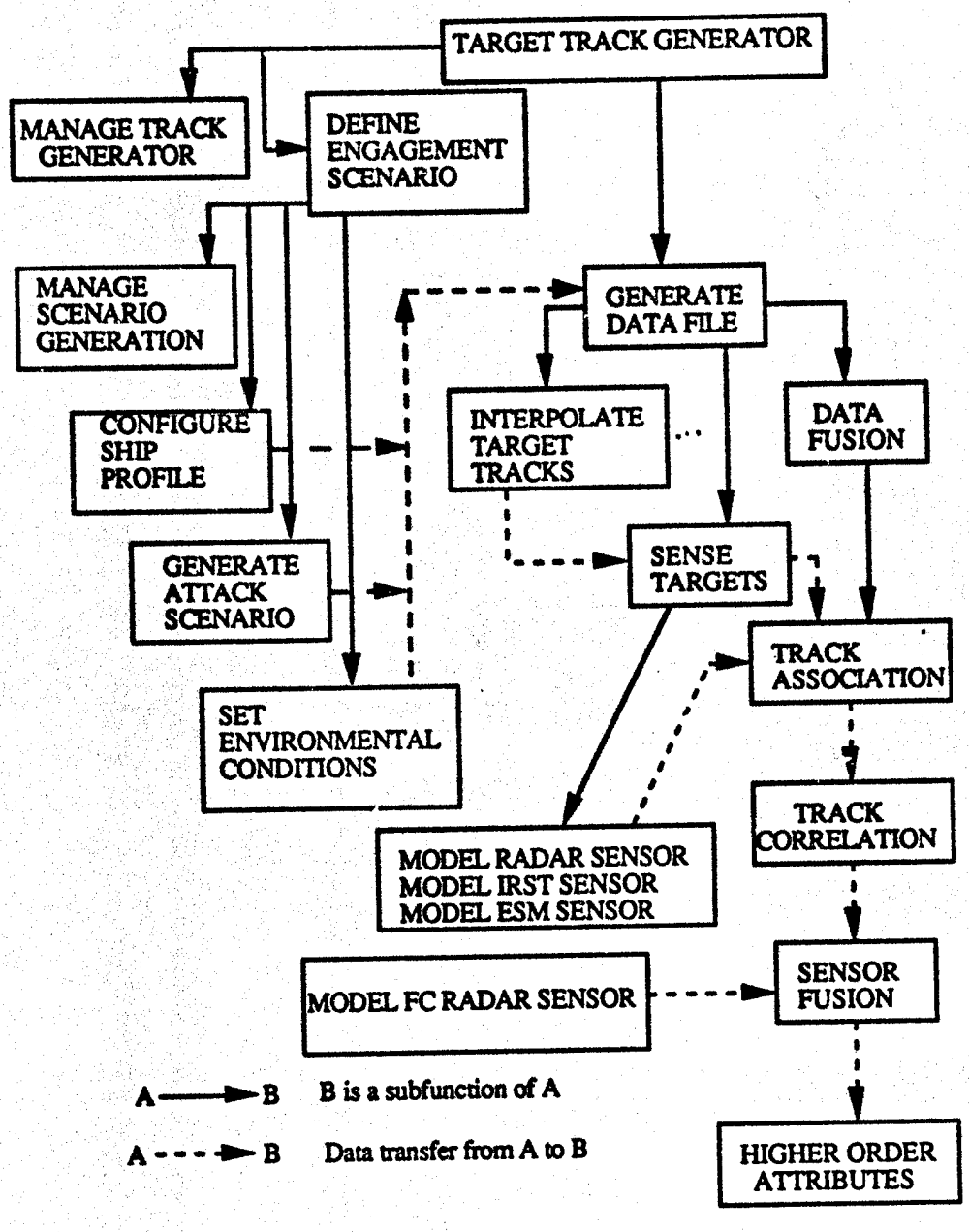


Figure 1. Target Track Generator

where A is called the conditional part and X the conclusion. Threat levels are based on track observables (as well as whether the track is currently under prosecution). Various

observable attributes of the track are interpreted, via the mechanism of rules, in order to determine the level of threat. In theory, the level of threat would be determined by associating a threat level with each unique combination of track observables. In practice, however, this is impractical because the number of combinations is very large.

Our approach, therefore, is to have a two-level strategy of rule application. At the lowest level, "raw" data is used to generate a limited set of hypotheses about various track attributes : identity, kinematics, radar state, and engagement status. For each of these four attributes, a set of rules is used to reduce the "raw" data into a manageable set of hypotheses specific to the attribute. These hypotheses then constitute the "data" to be used by the upper level set of rules. The output of this upper level is a set of propositions about the level of threat posed by the track. Each such proposition contains a statement chosen from the following list :

T/L=0 : Not a Threat. The track is believed to pose no risk to the Force. A hostile track can be a NT (non threat) if it is not believed to be capable of harming the Force. A "friend" also falls into this category.

T/L=1 : Threat Level = 1. A very slightly threatening track.

T/L=2 : Threat Level = 2. A slightly threatening track.

T/L=3 : Threat Level = 3. A moderately threatening track.

T/L=4 : Threat Level = 4. A threatening track.

T/L=5 : Threat Level = 5. A highly threatening track.

Rules to Generate Identity Propositions

The identity propositions are intended to quantify the inherent potential of the track. They quantify its destructive potential independently of its specific behaviour.

if: (trackId = 'Exocet')

then:

[identity is: 'moderateEnemy' withCertainty: 1.0].

Basically, in order to reduce complexity, we choose to categorize all track identities into one of 5 categories : Friend; Weak Enemy; Moderate Enemy; Strong Enemy; Unknown. There are four rules for testing track identity, one for each of the categories : weak enemy, moderate enemy, strong enemy and unknown. At the present time, scenarios involving the intermittent arrival of friendly and hostile units have not been planned for the TEWA.

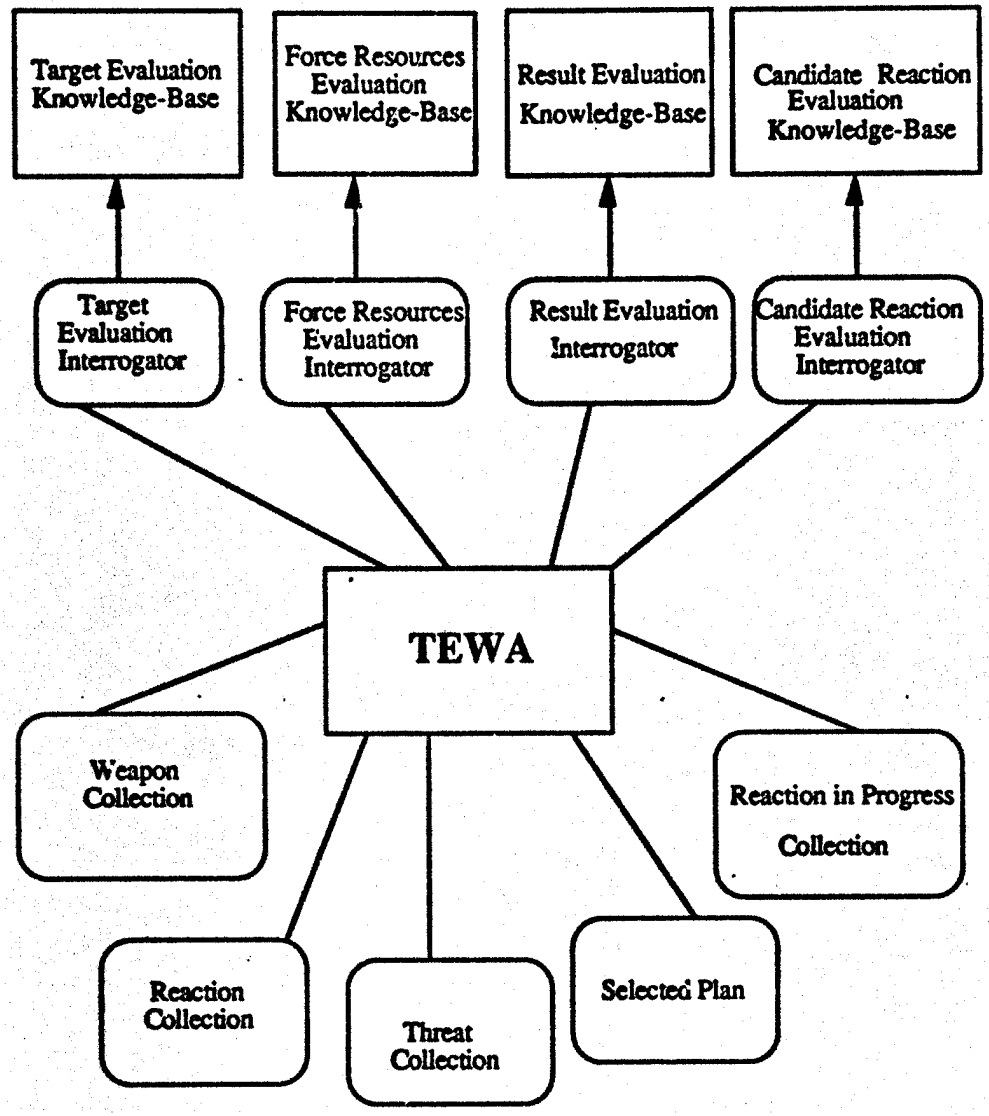


Figure 2. An Architecture for the Knowledge-Base TEWA

Hence, there are no rules identifying friendly units in this knowledge-base that can be used later with other rules to generate zero threat levels with an appropriate confidence factor.

Rules To Generate Kinematic Propositions

These propositions are intended to quantify the degree to which the kinematic behaviour of the track is threatening. In what follows, the CPA (closest point of approach) is the perpendicular distance in metres from the ship to the threat's direction in the horizontal plane and TTGI is the time in seconds for the threat to reach the CPA from its current position if it travelled at its current velocity. The rules to generate kinematic propositions are of the form :

if: (closing & (cpa ≤ 100) & (ttgi ≤ 40) & (ttgi > 20))
then:
[kinematics is:'strong' withCertainty: 1.0].

The kinematics classes are opening, strong, moderate and weak. There are thirteen rules of this type.

Rules to Generate Engagement Status Propositions

Whether or not the track is currently under prosecution does bear on the level of threat posed by the track. After all, a track that is under attack is less of a threat than the same track if it is not under attack. The rules to generate engagement status propositions are of the form :

if: (prosecutionStatus = 'notUnderProsecution')
then:
[engagementStatus is :'notEngaged' withCertainty:1.0]
else:
[engagementStatus is:'engaged' withCertainty:1.0].

There is only one rule in the knowledge-base of this kind.

Rules to Generate Propositions About Level of Threat

We now come to the highest level of rules. Instead of four rule sets, we have a single rule set. The propositions produced by rule firings at the lower level serve as the arguments of the rules at the highest level. The certainty factor associated with each rule will be assumed = 1.0. The certainty of the consequent will therefore be directly determined by the certainty of the antecedents.

In HUMBLE, the confidence factors associated with the parameters of a HUMBLE entity may take values going from -1 to 1. A value of +1 indicates that the system is absolutely certain that the value of the parameter is correct. A value of 0 indicates that the system is completely ignorant of what the value of the parameter should be. A value of -1 indicates that the system is absolutely certain that the value of the parameter is not correct. The format of HUMBLE rules is :

if: (A) then: [X withCertainty:CF].

If the conditional part A of this rule is true before firing (i.e., c(A) is one) and in what follows the certainty of X is denoted by c(X), the new calculated value of the certainty of X after rule-firing c(X|A) will be calculated as follows:

$$c(X|A) = c(X) + [CF*(1.0 - c(X))] \quad \text{if } c(X), CF > 0$$

$$= c(X) + [CF*(1.0 + c(X))] \quad \text{if } c(X), CF < 0$$

$$= (c(X) + CF)/(1.0 - \min\{|c(X)|, |CF|\}) \quad \text{if } c(X), CF \text{ are of opposite sign and not equal to } -1$$

If c(A) < 1 then the new certainty of X after rule firing c(X|A) is given by similar formulae depending on c(A), c(X) and c(F). In many cases, the conditional part of the rule is not simple. The rule can have a composite antecedent as follows:

if: (A and B) then: [X withCertainty:CF].

In this case, the certainty of A and B must be calculated before the rule can be fired. In HUMBLE, it is defined as follows:

c(A and B) = minimum of {c(A),c(B)}

Generally speaking, the confidence factors associated with the CPA and TTGI are calculated at the time that the sensor error estimates in position, bearing and velocity are calculated. The initial certainty associated with TTGI and CPA are defined by the formulae:

$$c(CPA) = \max[0, 1 - \text{sigma(CPA)}/|CPA|]$$

$$c(TTGI) = \max[0, 1 - \text{sigma(TTGI)}/|TTGI|]$$

where sigma(CPA) and sigma(TTGI) are the standard deviations of the CPA and TTGI respectively and can be calculated from the appropriate formulae for the CPA and TTGI.

The values of c(CPA), c(TTGI), CPA and TTGI are used to calculate the certainty of the conditional part of each kinematics rule and after the rule-firing, the certainty of the conclusion becomes known. This certainty can then be used in a rule about current track parameters such as :

if: (identity = 'strongEnemy' & (kinematics ='strong') & (engagementStatus = 'notEngaged'))

then:

[levelOfThreat is:5 withCertainty:1.0].

to compute the value of its antecedent. If the values of the other parameters making up the antecedents are not known, backward chaining must be used to obtain these values. Once these values are known, the rule may be fired to assign a threat level of five to the track with a certainty which is a function of the previous uncertainty and the current uncertainty of the antecedents. There are thirty two rules of this type in the knowledge-base.

In order to add a temporal aspect to the Threat Evaluation function consider the following upgrade to the functionality described above:

The computation of a threat ranking at $t=t_0$ will be a function of the threat rankings at two previous time steps:

$$FTL(t_0) = [\ 4*MTL(t_0) + 2*MTL(t_0\text{-}1) + MTL(t_0\text{-}2)\ ]/7$$

The term "FTL" stands for "Filtered Threat Level" while the term "MTL" stands for "Measured Threat Level". This represents a time-filtered approach to threat level generation. The threat ranking is mostly dependent on current threat parameters. However, there is some dependence upon past threat ranking. The computed value of $TL(t_0)$ should be rounded to the nearest whole number in order to be consistent with the way the TL values are utilized in other sections of the TEWA.

Now each of the MTL values will obviously have a certainty value. The certainty of the filtered threat level is calculated from the certainties of the measured threat levels of the current and two previous time steps by using the same formula as for the threat levels.

Rules based on Historical and Current Track Parameters

Under our current design, one and only one rule from the rule set based on current track parameters will fire for each track. We now look at how patterns of threat "behaviour" over time affect threat levels. The following rules concern manoeuvering targets or targets for which the estimated target velocity direction is subject to error. The following definitions are required to understand these rules :

Definitions :

- Let the line defined by the velocity vector of the track be "A".
- Let the line joining the track to the ship be known as "B".
- Let ß(A ->B) be defined as the angle measured clockwise swept out starting at A and moving to B.
- Let $ß(A \text{->} B, t = t_0)$ be defined as the value of ß measured at $t = t_0$.

Rules:

IF ((ABS(ß(A->B,t=t_0))<5) AND (ABS(ß(A->B,t = t_0-3))>10)

AND (TL < 4)) THEN

Reduce belief/confidence that TL < 4 and increase confidence that TL = 4.

More specifically, "divide" the existing certainty associated with the TL (that is<4) amongst the hypothesis that TL = 1,2,3 and that TL = 4. This may be confusing, so an example follows :

Suppose the FTL value for a threat is = 2 with C = 0.71. Further suppose the rule immediately above is satisfied. Under this condition, we should obtain the following levels of threat with associated certainty factors:

FTL = 2, C = 0.35

FTL = 4, C = 0.35

END IF

Further rules can be developed to cover the other orientations of a laterally manoeuvering target approaching a ship. The Candidate Reaction Evaluation knowledge-base must be informed of a manoeuvering target, because this will have an effect on determining whether it is engageable. The format of the rules in the Result Evaluation and Force Resources Evaluation knowledge-bases are similar in form to those of the Target Evaluation knowledge-base.

THE CANDIDATE REACTION EVALUATION KNOWLEDGE-BASE

The purpose of Candidate Reaction Evaluation is essentially twofold. Firstly, the process has to determine the engageability of each "surviving" track with respect to each weapon system. Secondly, it has to predict the effectiveness of each [threat-reaction] pair that survives the engageability determinations.

The ultimate purpose of CRE is to produce a set of reaction options. Each reaction option consists of

- A track to be prosecuted;
- A weapon system capable of prosecuting the track immediately;
- A prediction of the effectiveness of the use of the weapon against the track if weapon action is undertaken immediately.

Engageability Criteria

The number of reaction options reflects the number of different reaction-track combinations that are feasible (i.e.engageable). The specific algorithm for engageability determination of the SM-2/STIR system is given by the following rule:

if:((sm2Status='operational')&(stirAcanSeeTrackAtRollOver)&(sm2CanInterceptTrack)
&(stirA_Availability))

then:[engageability is:'engageableBySm2SalvoStirA'withCertainty:1.0.

engageability is:'engageableBySm2SingleStirA' withCertainty:1.0]

The antecedents of this rule are determined within the simulation framework. The criteria for establishing that a track is engageable by the [76 mm / STIR or LIROD] is defined by a similar rule. In order to ensure engageability, it is not sufficient that track interception occur inside the 76 mm gun's envelope. In addition, a STIR/LIROD fire-control radar must be capable of tracking the target. In our TEWA, we have made the following simplifying assumption : a test is made regarding whether the STIR can track the target at the reaction time. If this test is satisfied, we conclude that the STIR in question can track the target under consideration. The engageability rules for the CIWS gun system are similar to those of the 76 mm gun.

The following definitions and information are used in our specification of rules for softkill engageability. The threat radar mode can exist in three states : passive, search and track. The Deployment Time is the time interval from the decision to use the softkill weapon until the moment of softkill weapon activation. This is very short for the use of a jammer. For chaff, this represents the time from decision to use until the chaff canister explodes. The Time to Start Having an Effect is the time from softkill weapon activation until the weapon can begin having a measurable effect on the track. The Time Before Burnthrough is the time (measured from the decision to use a softkill weapon) until burnthrough occurs. Burnthrough occurs when the ship becomes the dominant target as far as the threat is concerned. We will assume for simplicity's sake that burnthrough will always occur at a specific range (threat-ship). The turn on time is the time (measured from the decision to use the softkill weapon) until the threat's seeker (radar) begins to receive a signal from the softkill weapon whose magnitude exceeds the minimum detection threshold. In simpler terms, the turn on time is the first point in time at which the threat can become aware of the presence of the softkill weapon.

The following must all be true in order for a track to be engageable by the jammer and/or chaff. Although not explicitly stated in "IF-THEN" format, these conditions can be

put together in order to create a rule for engageability (IF all conditions satisfied THEN track is engageable by this softkill weapon):

1. The softkill status (from Force Resources Evaluation) must be either "Operational" or "Degraded".
2. The track must be in either "Search" or "Track" modes.
3. The time to burnthrough must exceed the sum of the deployment time and the time to have an effect. Otherwise, it is too late to use the weapon.
4. The present time (this is the time at which the decision to use the jammer might be taken) plus the deployment time must exceed the turn-on time. Otherwise, it is too early to use the softkill weapon.

In addition, we have blind zone constraints with respect to the use of the jammer.

Effectiveness prediction for weapon options

There are two types of effectiveness measures associated with a particular [track,weapon] pair. First, there is the predicted effectiveness of the weapon in destroying / decoying the track. Secondly, there is the predicted impact of the reaction on the Force (e.g. risk of fratricide, reduced fighting capability, etc.). Note that the effectiveness prediction is only performed on those [track,weapon] pairs that pass the engageability test. All [track,weapon] pairs that fail the engageability test are not subject to any further consideration at the "present time".

Let the term Eff A stand for the predicted effectiveness of a weapon system with respect to the goal of destroying the track. Eff A is a function of two parameters; the type of the weapon and the type of the target. In order to determine Eff A for a given [track,SM-2] pair a table of probabilities is used. Similarly there are models to calculate Eff A, for the 76 mm gun, the close-in weapon system, the chaff system and the jamming system.

In order to predict chaff interference there are rules to determine whether or not the use of chaff will interfere with any line of sight to any track. Chaff reactions generally involve more than one chaff cloud. To simplify things, we said that if any chaff cloud is predicted to obscure any line of sight, then Eff B = 1 (i.e., there is an "impact on the Force"). We further assumed, for the purposes of this calculation, that all active sensors (STIRs + Phalanx + Search radars) are located at the centre of the ship. Finally, the prediction of chaff interference is only meaningful when the prediction is made with respect to a specific point in time or interval of time. To simplify, we will only examine interference at a single point in time. The point in time will be the deployment time (i.e., if the chaff clouds were

out there now, would there be obscuration ?). Under the limitations of the model, i.e., the ship does not move and under conditions of no wind, the ship's sensors will be effective in zones well clear of the clouds.

We now consider the determination of Eff C, a measure of how much the use of the weapon reduces the future fighting capability of that weapon. It will be assumed that Eff C will be set to 1 if the use of the weapon reduces remaining stock below some critical level (as determined by rules in the knowledge-base). Otherwise Eff C will remain = 0.

RANKING OF CANDIDATE REACTIONS

The purpose of this function is to generate a plan of action for dealing with threats. We have threat evaluations, resource evaluations, and result evaluations. We have [track,weapon] pairs where the track is engageable by the weapon and where associated effectiveness values have been established. Our task is now to generate a plan of action based on all this information. In order to generate a plan, we need to establish the goal (or goals) of our plan. The highest level goal is to protect ourselves and our consort from destruction. Our plan should be the one that gives us the highest probability of survival. Another goal is to conserve resources. Intuitively, this goal is less important than the goal of survival. Another less important goal is to minimize "impact on the Force"(interference). The ranking of plans is accomplished by an optimization function in the TEWA object of the simulator (see figure 2). In what follows, T_i is a track identifier. We will assume that there are "N" tracks in the system at any time. Let C(i,0) be the confidence that track i is a friend, C(i,1) be the confidence that track i is at Threat Level = 1 and $\Omega(i)$ be the uncertainty associated with determining the threat ranking of track T_i. In addition, let $[T_i,W_k]$ be a track-weapon combination where weapon W_k is capable of engaging track T_i and let Eff_A(i,k), Eff_B(i,k), Eff_C(i,k) be used to refer to the Eff_A, Eff_B, Eff_C values associated with a $[T_i,W_k]$.

Plan Scoring Using an Optimization Function

Survival points are calculated for each track T_i that has at least one weapon assigned to it in the proposed plan by computing the value of the following function:

$$1 * (Eff\_A(i,*)*C(i,1)) + 2 * (Eff\_A(i,*)*C(i,2)) + 3 * (Eff\_A(i,*)*C(i,3)) +$$

$$4 * (Eff\_A(i,*)*C(i,4)) + 5 * (Eff\_A(i,*)*C(i,5)) + 3 * (Eff\_A(i,*)*\Omega(i)).$$

Denote this value by F_1. The "*" character is used in order to reflect the fact that there may be more than one weapon
assigned to track T_i. We desire plans that try to ensure that all threats are subject to prosecution. More, specifically, we wish that any plan that deals with n+1 tracks be deemed superior to any plan that deals with n tracks regardless of the F_1 values associated with the plan. This can be accomplished by computing our final survivability result as follows :

$$\text{Survivability} = (F_1 + n)/(N + 1)$$

where n = number of tracks in the plan;
N = number of tracks submitted to TEWA.

Each [track(i), weapon(k)] pair associated with a plan will have an Eff_B parameter associated with it. If Eff_B = 1 then the use of weapon(k) will produce interference. A quantitative measure of the amount of interference associated with a plan is given by the following :

$$\frac{\text{Number of [track(i), weapon(k)] Pairs where Eff\_B} = 0}{\text{Total Number of [track(i), weapon(k)] Pairs}}$$

If every [track(i), weapon(k)] combination yields interference, this value will = 0, if half produce interference the value will equal 0.5, and if no combination yields interference, the value will equal 1.0. The higher this value the better. Denote this ratio by Impact.

Each [track(i), weapon(k)] pair associated with a plan will have an Eff_C parameter associated with it. If Eff_C = 1 then the use of weapon(k) will produce resource usage problems. A quantitative measure of the extent of resource usage problems associated with a plan is given by the following :

$$\frac{\text{Number of [track(i), weapon(k)] Pairs where Eff\_C} = 0}{\text{Total Number of [track(i), weapon(k)] Pairs}}$$

If every [track(i), weapon(k)] combination yields resource usage problems, this value will = 0, if no combinations yield resource usage problems, the value = 1.0. Denote this ratio by Res. Usage. The proposed optimization function F for scoring a given plan is as follows:

F(Plan) = 0.8 * (Survivability) + 0.1 * (Impact) + 0.1 * (Res. Usage)

The plan with the highest score is implemented immediately.

RESULTS FROM KNOWLEDGE-BASE TESTING SCENARIOS

The knowledge-base TEWA for a TRUMP-like ship was tested in the scenario shown in figure 3.

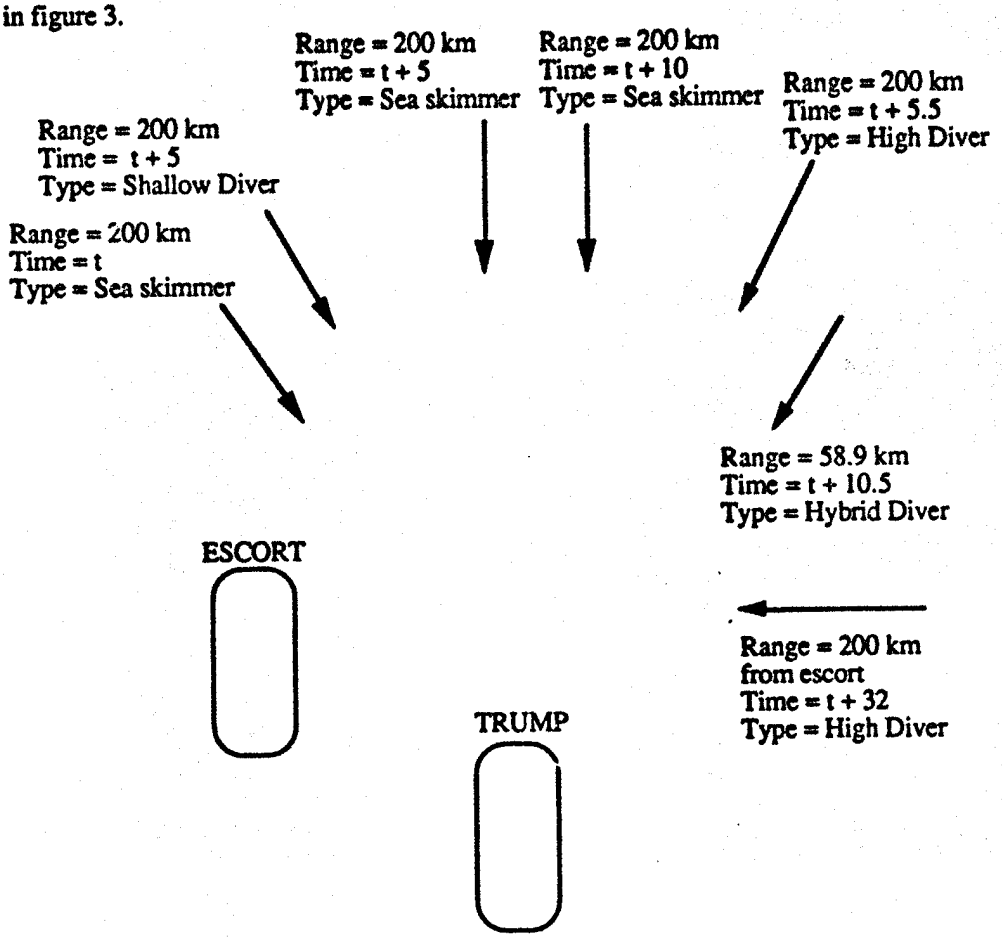


Figure 3. A Testing Scenario for the Knowledge-Base TEWA

In this scenario, six targets are directed at the TRUMP-like ship and one at the escort. The sequence of weapon assignments performed by the knowledge-base as well as the targets

destroyed are shown in figure 4. It is to be noticed that three different weapons are assigned to the first high diver. The SM-2/STIRB fires and misses this target. However, the system still remains assigned to it. As this target flies into the ship, the gun/STIRA and CIWS systems are respectively assigned to the high diver. Thus, after the kill assessment for the first five targets is complete, three weapon systems are assigned to the sixth target because the seventh target is not engageable by any of these systems. The gun/STIRA finally destroys the sixth target. In this testing scenario, therefore, six targets are destroyed and one missile hits the escort. The knowledge-base TEWA was also tested in a scenario with five threats and in this case neither the TRUMP-like ship nor the escort was hit by a missile.

	THREATS						
WEAPON / RADAR	HYD	SS1	SS2	SD	SS3	HD1	HD2
SM-2/STIR A	KILL	KILL		KILL			
SM-2/STIR B			KILL			*	
GUN/STIR A						*	
GUN/STIR B							
GUN/LIROD					KILL		
CIWS						*	

SM-2 = Standard Missile 2 CIWS = Close-in Weapon System

LIROD = Gun radar STIR A,B = Radars

* = indicates an assignment of weapon/radar to threat

Figure 4. Snapshot from the Knowledge-Base Testing Scenario

22

REAL-TIME CONSIDERATIONS

The knowledge-base TEWA which has been described above does not function in real-time. The field of real-time knowledge-base systems is a field of current research which has not fully developed yet. The author has read the descriptions of three real-time systems in the open literature. These are known as Pilot's Associate, the Adaptive Suspension Vehicle and the Autonomous Land Vehicle. They are described in references [12] and [14]. The Pilot's Associate is a knowledge-base system which attempts tactical situation assessment and weapon assignment for fighter aircraft In reference [12], a methodology is given to make it function in real-time. The real-time version of Pilot's Associate has been tested in various environments to assess its performance, task scheduling and real-time interaction with a realistic environment. The real-time version of Pilot's Associate is expected to be functional by 1992.

The author has analyzed some of the real-time requirements for the simplified naval TEWA problem, i.e., that of a single warship attacked by anti-ship missiles. A preliminary survey shows that the total time for rule-firings and backward chaining in the Target Evaluation knowledge-base of figure 2 should be compatible with the data output rate of the ship's sensor fusion process. This data rate will depend on the type of sensor fusion chosen. In the case of the Result Evaluation knowledge-base, the total time for rule-firings should be compatible with the weapon system for which it is performed. Thus in the case of missile systems, this time must be compatible with the illumination time of the fire-control radar to establish that the track has been destroyed. In the case of a softkill weapon system, it must be compatible with the amount of time required to establish that the track has been successfully decoyed. The author sees that the status, availability and stock level of weapon systems as estimated by the Force Resources Evaluation knowledge-base should be made known to the blackboard controller as soon as possible, i.e., the fastest possible rule-firings are necessary to make sure that the central unit of the TEWA is aware of the new ship situation. A preliminary analysis shows that if the total time to perform engageability calculations, effectiveness predictions and ranking of plans is compatible with the data output rate of the sensor fusion process, a new reaction can be selected each time the tactical situation changes.

At the present time, the author has not studied the scheduling of each one of the above tasks : (Target Evaluation, Force Resources Evaluation, Candidate Reaction Evaluation, Ranking of Candidate Reactions) with respect to the central blackboard. It is

hoped that an efficient task scheduler can be found to execute these tasks as rapidly as possible. If this is not the case, some work must be done to have the tasks executed concurrently so that the above requirements can be satisfied.

These ideas constitute a first glance of real-time knowledge-base systems for naval Command and Control and they could be subject to change.

CONCLUSIONS

1)The performance of the knowledge-base TEWA was examined in several anti-air warfare testing scenarios and found to be satisfactory. The performance of a TRUMP-like ship with a conventional TEWA was tested in similar scenarios (but with fewer threats) using a validated Ship Combat System Simulation and found to have comparable results with the ship having a knowledge-base TEWA.

2)It would appear that the knowledge-base approach to the Tactical Situation Evaluation of a warship under attack by anti-ship missiles is a reasonable one. Some of the problems to be overcome in this field are identification of air threats, estimation of threat kinematic parameters, evaluation of threat seeker head behaviour and prediction of the behaviour of manoeuvering targets. This study shows that these areas could be reasonably handled by the knowledge-base approach. A reasoning approach whereby information from kinematic, radar state and identity parameters are put together provides a good tactical picture for weapon assignment. References [1] to [5] are looking at the applicability of knowledge-base systems to Tactical Situation Evaluation.

3)The current form of the knowledge-base TEWA was found to be adequate for a single ship equipped with radar guided weapon systems, surveillance radars, an electronic support measure system and an infrared search and track device. There was a debate as to the suitability of the knowledge-base approach in doing Threat Evaluation and Weapon Assignment for various kinds of hardkill and softkill weapon combinations. In order to accommodate a laser guided hardkill weapon or a different kind of softkill weapon in the knowledge-base TEWA, it would only be necessary to change the rules concerning the use of these systems in the appropriate knowledge-base and integrate the latter to the functions of the platform.

4)The tactical situation around the ship is perceived through its sensors which measure parameters relating the target to the ship. Radar sensors, for example, will measure the slant range and bearing of a target and will return values which are subject to errors. In the target evaluation knowledge-base, a beginning has been made with Mycin type uncertainty

to obtain a consistent set of threat levels from kinematic, radar state and threat identity parameters. Although the results obtained were satisfactory, there is no guarantee that this is the best method for dealing with sensor data uncertainty.

5)The knowledge-base TEWA that we have built is not a real-time system. In order to make it execute in real-time, separate studies would have to be done to choose an appropriate computer architecture and computer software to implement these ideas in a real-time knowledge-base TEWA. The preceding section presents some ideas that must be addressed in a real-time naval knowledge-base system.

ACKNOWLEDGEMENTS

The author would like to acknowledge the contribution of Thomson CSF Systems Canada in this exploratory work to apply knowledge-base methods to the Threat Evaluation and Weapon Assignment process of an AAW destroyer or frigate.

REFERENCES

[1] Byrne, C.D., Miles, J.A.H. and Lakin, W.L., "Towards Knowledge-Based Naval Command Systems", IEE Proc.3rd Inter. Conf. C^3MIS, Bournemouth, England, 1989.

[2] Anderson, B.A., Lakin, W.L. and Byrne, C.D.,"Operational Use of Knowledge-Based Military C^2 Systems", IEE Proc.3rd Inter.Conf. C^3MIS, Bournemouth, England, 1989.

[3] Montgomery, J.D. and Byrne, C.D., "Evolving the User Interface for Knowledge-Based Command Systems", IEE Proc.3rd Inter.Conf. C^3MIS, Bournemouth, England, 1989.

[4] Hirst, R.A.,"Allegiance Assessment Using Explanation Based Reasoning", IEE Proc. 3rd Inter.Conf. C^3MIS, Bournemouth, England, 1989.

[5] Miles, J.A.H., "Architectures for C^2 Knowledge-Based Systems", IEE Proc.3rd Inter.Conf. C^3MIS, Bournemouth, England, 1989.

[6] Sriram, D. and Rychener M.D., "Expert Systems for Engineering Applications", IEEE SOFTWARE, March 1986, pp. 3-5.

[7] Thompson, T.F., and Clancey, W.J., "A Qualitative Modeling Shell for Process Diagnosis", IEEE SOFTWARE, March 1986, pp. 6-15.

[8] Ramana Reddy, Y.V., Fox, M.S., Husain, N. and McRoberts, M., "The Knowledge-Based Simulation System", IEEE SOFTWARE, March 1986, pp. 26-37.

[9] Kim, J. and McDermott, J., "Computer Aids for IC Design", IEEE SOFTWARE, March 1986, pp. 38-47.

[10] Hewish, M. and Turbé, G., "Naval decision-support aids, Part 2: non-US systems", DEFENSE ELECTRONICS & COMPUTING (SUPPLEMENT TO INTER. DEF. REVIEW), May 1991, pp. 57-58.

[11] Hewish, M., "Naval decision-support aids, Part 1 : US systems", DEFENSE ELECTRONICS & COMPUTING (SUPPLEMENT TO INTER. DEF. REVIEW), March 1991, pp. 14-21.

[12] Banks, S.B. and Lizza, C.S., "Pilot's Associate. A Cooperative, Knowledge-Based System Application", IEEE Expert, June 1991, pp. 18-29.

[13] Waterman, D.A., "A Guide to Expert Systems", Addison-Wesley Publishing Company, Reading, Massachusetts, 1986, pp. 66-69.

[14] Payton, D.W. and Bihari, T.E., "Intelligent Real-Time Control of Robotic Vehicles", Communications of the ACM, Vol.34, No.8, August 1991, pp. 49-63.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

THE CF-18 DIAGNOSTIC AND MAINTENANCE EXPERT SYSTEM PROJECT

J. B. Tubman, M. D. Brinsmead[1], C. C. Lumb[2], M. M. Mendoza[1], and I.S.L. Tan[1]

Abstract

The goal of this project was to apply AI technology to diagnosis and maintenance of CF-18 aircraft. The domain chosen was throughput increase for the AN/USM-469 radar test set for the AN/APG-65 radar, with the radar transmitter as unit under test. Historical records were maintained about symptoms, tests, and probable causes of failure. These were used to compute a new, faster test sequence, subject to certain safety constraints. The system is being field tested at the CFB Cold Lake RTS laboratory, and the results of the field test will be analysed in mid- to late-1992. Some preliminary conclusions were reached before this field trial and will be discussed here.

[1]Research Officer, [2]Program Manager, Alberta Research Council, Advanced Computing Engineering Dept, Calgary

INTRODUCTION

The maintenance and repair of the various sorts of aircraft operated by the Canadian Armed Forces is a demanding and important task. Large numbers of highly skilled technicians are required to keep today's sophisticated aircraft ready for flight. The Alberta Research Council (ARC) approached the Chief of Research and Development (CRAD) with the idea of applying artificial intelligence (AI) and expert system technologies to problems related to the diagnosis and maintenance of CF-18 aircraft. The CF-18 expert system project (CFES-18) was undertaken with the main objective of investigating the feasibility of applying AI and expert systems to CF-18 avionics maintenance. It was hoped that the application of these technologies would lead to cost savings, increased safety, and improved flight readiness.

Diagnosis and maintenance of aircraft is a large domain, and it was necessary to select some relevant subset of the problem. The first phase of the project was devoted to this task. In the end, it was decided that an appropriate problem would be using artificial intelligence and expert systems to increase the throughput on an existing piece of automatic test equipment, namely the AN/USM-469 radar test set (RTS). The RTS is used to test an AN/APG-65 radar on the CF-18 aircraft. Phase II involved performing an analysis of the problem area, and deciding on the functionality requirements and high level design of the system which we have called the Knowledge-based Adaptive Test Sequencing System (KATSS). The last phase involved performing a detailed analysis and design towards producing and field testing a prototype. It was determined that for the prototype, we would concentrate on improving the throughput only when testing the radar's transmitter.

This paper reports on the results of this project, including descriptions of the problem area, the approach taken towards a solution, and lessons we have learned so far in the project.

SELECTION OF THE PROBLEM

The CFES-18 project is a feasibility study intended to investigate the appropriateness and practicality of applying expert system technologies to the maintenance of CF-18 aircraft. Initially, the intent was to develop a protocype expert system for fuel system maintenance. However, this problem was eventually solved by other means, which gave us the opportunity to investigate some other aspect of the aircraft's maintenance.

We began by examining the current situation with regard to CF-18 maintenance. We spoke with people in the Directorate for Fighter and Trainer Engineering and Maintenance (DFTEM), the Directorate for Aerospace Support Engineering (DASENG), the National Research Council Laboratory for Intelligent Systems, the New Shipboard Aircraft programme, Amtek Testware, and the Alberta Electronics Test Centre. A wide range of possible projects were suggested, including ones that were not expert systems.

Five alternatives were proposed to DND. These were: an expert system to augment the ATE for a small instrumentation subsystem; an unspecified expert system for another aircraft such as the CF-5 or the Sea King helicopter; trend analysis of maintenance data from the AMMIS or DMS data systems; trend analysis of engine data, to detect adverse trends and anticipate engine faults; and an automated technical manual system that would put Canadian Forces Technical Orders (CFTOs) on hypertext. DND chose the expert system for augmenting the ATE.

Since this was still a rather large domain, it had to be narrowed down further. With the help of the people in DFTEM, the area that was finally selected was increasing the throughput of the AN/USM-469 Radar Test Set (RTS), during the diagnosis of problems with the transmitter of the CF-18's AN/APG-65 radar. It was felt that the existing software on the RTS went through its test sequence in a very straightforward, unintelligent way, and that the throughput on the machine might be increased if the test order was redone in a way that took past experience into account.

THE AUTOMATIC TEST DOMAIN

Aviation electronics, or *avionics*, is an integral part of day-to-day flight operations. Complicated electronic devices for navigation and communication provide capabilities such as maintaining in-flight positions, pinpointing destinations, and landing safely. Therefore, the maintenance of these electronic systems is a significant aspect of the aeronautics industry.

Avionics maintenance involves different levels of complexity in diagnosis and repair, ranging from replacement on the aircraft of the larger assemblies of electronic boxes, to major repair of the smaller electronic cards at the supplier's electronics laboratory. Maintenance levels involve different types of test equipment. *Built-in test* (BIT) indicate whether a unit is faulty and needs replacement [1]. *Automatic test equipment* (ATE) is used to test a failed unit further to determine which of its sub-assemblies has caused the failure. ATEs are used at mid-level maintenance shops where the isolated sub-assemblies are replaced and may be sent for major repair. A unit which is replaced on the aircraft is normally referred to as a *weapon replaceable assembly* (WRA), while a sub-assembly replaced at maintenance facilities is a *shop replaceable assembly* (SRA). In brief, BITs diagnose down to the WRA level, and ATEs diagnose WRAs down to the SRA level.

Avionics maintenance is performed by highly skilled technicians who have an intimate understanding of the different electronic assemblies, and expertise in operating the testing equipment and in interpreting testing results. The ATE is an important tool for these technicians. It relieves them from the tedious and time consuming tasks of instrument set-ups, calibrations, plug/cable changeovers, and taking readings. The ATE performs such tasks on the *unit under test* (UUT) that is connected to it by automatically directing a sequence of tests. The tests are computer controlled through a test program which is often written in the ATLAS language (Abbreviated Test Language for All Systems).

For the CFES-18 project, the target of application is the two ATE stations within the CF-18 avionics maintenance facility of the Canadian Armed Forces at Cold Lake, Alberta. These stations house two radar test sets (RTS), designated AN/USM-469, for testing and fault isolating WRAs of the CF-18 AN/APG-65 radar. An RTS consists of racks of testing equipment, connectors, and an HP-100 computer and test programs, which control the equipment through a general purpose interface bus (GPIB, or IEEE-488). There are 72 different types of WRAs that can be connected to the RTS as UUTs, and we are concentrating on the radar transmitter for this project.

Each type of WRA requires a different set of test procedures, as detailed in the *test requirement document* (TRD) of the WRA. Software written in ATLAS encodes the test procedures for execution on the RTS computer, and is referred to as the Test Program Instructions (TPI) for the WRA. Technicians control the execution of this test program and the contained test procedures through a command program called the Test Executive [2]. The Test Executive allows the user to specify the statement number in the test program where testing will start (Start At Entry point, or SAE), and a statement number where testing will stop (Halt At Statement, or HAS), if any. UUT testing may be carried out by supplying a series of SAE-HAS pairs until the RTS has stopped to indicate a fault, or all the required test procedures have been executed. Normally, however, the technicians would simply supply the very first SAE for the UUT, to perform the tests end-to-end rather than in groups.

The test program for the radar transmitter [3] contains a sequence of at least 160 test procedures partitioned into 14 groups [4]. For safety reasons, the first eight test groups must be performed in the given sequence, and the last six test groups may be re-ordered. A job order for transmitter testing is specified by personnel who replaced the transmitter, on a Maintenance Form CF543 attached to the UUT. The form may include descriptions of the WRA failure, and fault codes produced by the BIT. Experienced ATE technicians may be able to use this information to decide on the test procedure that will most likely find the faulty

SRA, and the test group to execute first in the re-ordered test sequence. The purpose of KATSS is to assist the technician in making this sequencing decision. The following section details the approach taken to accomplish this task.

APPROACH

The Resequencing Problem

The resequencing problem is an optimisation problem in which the goal is to rearrange a series of test operations to minimise the expected time required to detect a fault. For the purposes of this discourse, we shall describe two classes of resequencing problem: the *general resequencing problem;* and the *special resequencing problem.* The two classes are distinguished entirely by the way in which expected duration is computed.

In each case, a number of common assumptions are made regarding the nature of the test operations which are to be resequenced. The common assumptions shall be presented first, to be followed by a description of the special assumptions differentiating the two classes of sequencing problem.

Assumption 1:

Test operations may be decomposed and described hierarchically.

This is certainly the case for the test operations described in the TPI for the AN/USM-469 Radar Test Set (RTS), upon which our research was based. The tests, as described in the ATLAS source code are organised into several levels of hierarchy: Test Blocks, Test Groups, and Test Steps. For our purposes, a Test Step is the fundamental unit of test opera-

tion; a Test Group[3] is an ordered set of one or more Test Steps, and a Test Block is a set (ordered or unordered) of Test Groups and/or other Test Steps.

Assumption 2:

Test operations are essentially independent of one another. That is, the instructions executed by the Automated Test Equipment in the performance of one test operation are independent of any test operations which may be rescheduled relative to the test operation in question.

Again, this is (for the most part) the case for the AN/USM-469 Radar Test Set. This assumption gives rise to the need for constraints on the resequencing of test operations. For example, within a 'Test Group', it is frequently the case that one Test Step may define state information which will modify the behaviour of a subsequent Test Step within the same Test Group.

If this is to occur, these Test Steps *must* be constrained to be executed in a fixed and predetermined order. In the case of our test system, Test Groups are by definition *static*, that is, the test steps within each test group can never be resequenced with respect to one another.

Assumption 3:

Test operations at a given level in the hierarchy are subject to certain constraints restricting how they may be reordered. These constraints are, however, relatively simple, and do not cross levels of the hierarchy.

---

3 For our purposes, a 'Test Group' is a set of TPI instructions delineated by entry points.

The primary source of constraints on resequencing tests is the need to assure that safety tests are performed before any actual performance tests are commenced. In the case of the RTS, several groups of safe-to-turn-on tests must be executed in a fixed sequence before any of the performance tests are executed. In support of this type of limitation, a simple constraint system which takes advantage of the hierarchical description of the test program was developed. This constraint system is described in the following section.

Assumption 4:

For each test operation, either the probability of detecting a fault is known a priori, or it can be reliably computed from any test operations contained by the one in question.

For each atomic test operation, the probability of finding a fault must be known prior sequencing the tests. In the prototype system developed by the Alberta Research Council, these probabilities are provided by a database module which estimates based upon past history and the symptoms reported for the UUT currently under consideration.

For composite test operations (test operations which contain one or more sub-operations), satisfaction of this assumption is assured by Assumption 2. Since no test operation may be contained by more than one "parent", it is always possible to recursively compute the probability of a composite operation detecting a fault based upon the probabilities of its constituent test operations. It is possible to demonstrate that the probability of detecting a fault in a sub-sequence of tests is independent of the order in which the operations are performed.

Assumption 5:

Atomic test operations (test operations at the lowest level of the hierarchy) have a fixed duration, and will always run to completion without regard to whether or not they detect a fault.

Although this is not strictly true, it provides a very convenient and powerful abstraction, especially when considering the special resequencing problem. In actual fact, the duration of a Test Step (recall that "Test Steps" are the Atomic Test Operations) may vary *slightly* depending on whether a fault is found. That is, the Test Steps will always run to completion regardless of whether a fault is found, but must take time to print an error message (or even sometimes resolve an ambiguity) when a fault is found. The minor deviations incurred when a fault is detected are not considered significant.

Assumption 6:

All testing is immediately halted when the first fault is detected.

This assumption is absolutely crucial to the functions described below for computing the expected duration of a test sequence. This is because we assume that after a fault is detected no additional tests need be executed, and that the duration of a test which is not executed is zero.

The Special Resequencing Problem

The special resequencing problem is characterised by the assumption that test operations are essentially self-contained and independent of one another. Under this model, we presume that the expected time required to perform a particular test operation depends solely upon the

"normal" duration of the operation, and the probability with which we can expect it to detect a fault.

Based upon this assumption, the expected time T_S to execute n test operations arranged on a sequence $O_0, O_1, O_2, \ldots, O_{n-1}$ is given in equations (1) and (2) below:

$$T_S = t_0 \tag{1}$$

where

$$t_i = D_i + (1 - P_i)\, t_{i+1} \tag{2}$$

and where D_i is the expected duration of the test operation in the i^{th} position in the sequence, and P_i is the probability of the test operation in the i^{th} position in the sequence detecting a fault.

and where

$$t_n = 0$$

Given this description of cost (duration) of a sequence of test operations, it is possible to define a '<' relationship as follows:

$O_i < O_j$ iff a sequence of test operations costs less when O_j follows O_i than when O_j precedes O_i in an otherwise unchanged sequence, that is, iff starting from a sequence where O_j precedes O_i and exchanging their positions results in a decrease in cost. (3)

Furthermore, it is possible to demonstrate that, given the cost function (1), the following is true:

$$O_i < O_j \quad \Leftrightarrow \quad P_i D_j < P_j D_i \tag{4}$$

and, also

$$\forall i \forall j \forall k \qquad O_i < O_j , O_j < O_k \Rightarrow O_i < O_k \tag{5}$$

These results (the derivation is omitted due to extreme ugliness), show us that the '<' relation is both transitive and independent of any other test activities in a sequence. Thus the special resequencing problem is equivalent to sorting a list of numbers, and can be solved optimally in $O(n \log n)$ time.

The General Resequencing Problem

The general resequencing problem is distinguished from the special resequencing problem by the assumption that test operations are not entirely independent from one another, and that there exists a certain natural ordering to the operations. Under this model, we presume that the expected time required to perform a particular test operation depends not only upon the "normal" duration of the operation, but also upon any penalties which may be levied for deviating from the "natural ordering" of the test operations.

That is, given a set of n test operations $O_0, O_1, O_2, \ldots O_{n-1}$ we assume that test operation O_i can only be executed without penalty only when it is executed immediately following O_{i-1}. In the case of the RTS, these penalties take the form safety tests which must be executed when starting a block of tests from an entry point, but which are not executed if the same block of tests is started as a result of completing the test block which immediately precedes it in the TPI.[4]

Based upon these assumptions, the expected time T_G to execute n test operations arranged in a sequence $O_{s0}, O_{s1}, O_{s2}, \ldots, O_{s(n-1)}$ is given in equations (6) and (7) below:

---

[4] The test operations defined by the TPI are normally executed sequentially in the form of an end to end test. At the end of each block of tests is a goto statement which jumps around the first few statements in the beginning of the next block of tests. These few statements are intended to re-execute the safe-to-turn-on tests when a test block is manually run by the operator.

$$T_G = t_0 \tag{6}$$

where

$$t_i = P(\sigma^{-1}(i-1), \sigma^{-1}i) + D_{\sigma i} + (1 - P_{\sigma i})\, t_{i+1} \tag{7}$$

and where σ is a permutation function mapping the set of integers from 0 to $n-1$ onto integers from 0 to $n-1$

and where

$$P(i,j) = \begin{cases} 0 & \text{if } i<0, \text{ or} \\ 0 & \text{if } O_i \text{ is the test operation which "naturally" precedes } O_j, \text{ or} \\ \text{Penalty}(j) & \text{otherwise} \end{cases} \tag{8}$$

and where Penalty(k) is the time penalty incurred for test operation O_k if it is executed anywhere other than immediately following test operation O_{k-1}

and where D_i is the expected duration of the test operation in the i^{th} position in the sequence, and P_i is the probability of the test operation in the i^{th} position in the sequence detecting a fault.

and where

$$t_n = 0$$

Unlike the special [illegible]cing problem above, the presence of the penalty terms prevents us from defining a simple '<' relation which provides an ordering relation on the test operations. We believe the solution to the general resequencing problem to be $\mathcal{NP}$-Hard, although we have not attempted to prove it.

Sequencing Constraints

As mentioned in the section describing the assumptions relating to the test sequencing problems, we are required to consider a number of limitations on the order in which test operations can be executed. The main source of these limitations for the AN/USM-469 Radar Test Set was the existence of several groups of safety tests, which must be executed in a fixed order prior to commencing any other group of tests.

These limitations easily could have been accommodated by implementing a test sequencer in such a way as to leave the safety tests untouched, and only consider for resequencing those tests which appear later in the TPI. It was our choice, however, to consider the possibility of somewhat more complex constraints when designing our system.

Although it was possible to provide a very general form of constraints, such as Allen's temporal logic [5], we opted for a simpler formalism which would not be more computationally complex than the sequencing problem. The formalism we selected was based upon the organisation of the activities in a test plan into a number of hierarchical blocks, and then designating each block as either static or dynamic. A static block is one whose contents must be executed in a fixed order, while a dynamic block is one whose contents may be executed in any order. A static block may contain a dynamic block, or vice-versa; the constraints apply only at the current level in the hierarchy, they are not inherited by contained blocks.

This form of constraint allows the specification of a wide variety of sequencing constraints[5], but is sufficiently restrictive that satisfaction of the constraints is still (trivially) easy.

---

[5] Certainly, a much wider variety that was strictly required for the task of resequencing tests for the AN/APG-65 radar transmitter.

Design and Implementation of the Test Sequencer

The test sequencer was designed to run as an independent program which would read an input file describing a set of test operations and their constraints, and which would generate an output file describing the order in which the test activities should be executed. This program is invoked and executed upon demand by the database component of the software, which also provides the user interface.

Earlier in this paper, we described two classes of sequencing problem. These two classes correspond to two main modes of operation of the RTS. The general sequencing problem corresponds to the normal use of the RTS in which operators controlling the program manually are permitted to start execution only at entry points, which forces the safety tests to be re-run for every new entry point. The special sequencing problem reflects a hypothetical mode of operation in which operators are permitted to execute the test program starting at any arbitrary statement number, thus (assuming the statements are correctly selected) allowing the safety tests to be skipped.

In order to permit the most flexible possible analysis of the test sequencer, its implementation includes facilities to support both modes of operation. The default mode is the "Start At Entry point" mode in which general resequencing is performed; the "Start At Statement" mode may be specified optionally.

The implementation of the special sequencing mode was done using quicksort [6]. A number of heuristics were tried for the general resequencing problem. The heuristic finally selected was a trinary bubble sort [6] which was demonstrated to exhibit high performance while almost always achieving optimal (and always near-optimal) results for problems the size of RTS. This can be extended to an n-ary bubble sort with a computational complexity of $O(n!\,(m^2 - mn))$ where m is the number of test operations to be sequenced.

Design and Implementation of the Database

The database component of KATSS stores and maintains information used in test sequencing. This includes testing information for each type of WRA (e.g., required test groups and procedures, and SAE, MAS, and typical duration for each test), and data recorded from each test activity on the RTS (e.g., UUT information, BIT codes, test number of the procedure where the RTS detected a fault, and KATSS-generated test sequence).

The database application functions, called the Data Collector, are mainly for accepting test activity type data from the user and storing this data in the database. These are:

- *Test Activity Initiation.* The user provides KATSS with the serial number of the CF543 attached to the UUT being tested on the RTS, and some information from this form such as the UUT part number and serial number, and BIT code and failure descriptions indicating the problems with the UUT. The test sequencer is then executed and any resulting sequence is stored in the database.

- *Test Sequence Display.* The user views the test sequence that KATSS has generated for a test activity.

- *Entry of Fault Detection.* The user enters the statement number displayed by the RTS when it stopped to indicate fault detection.

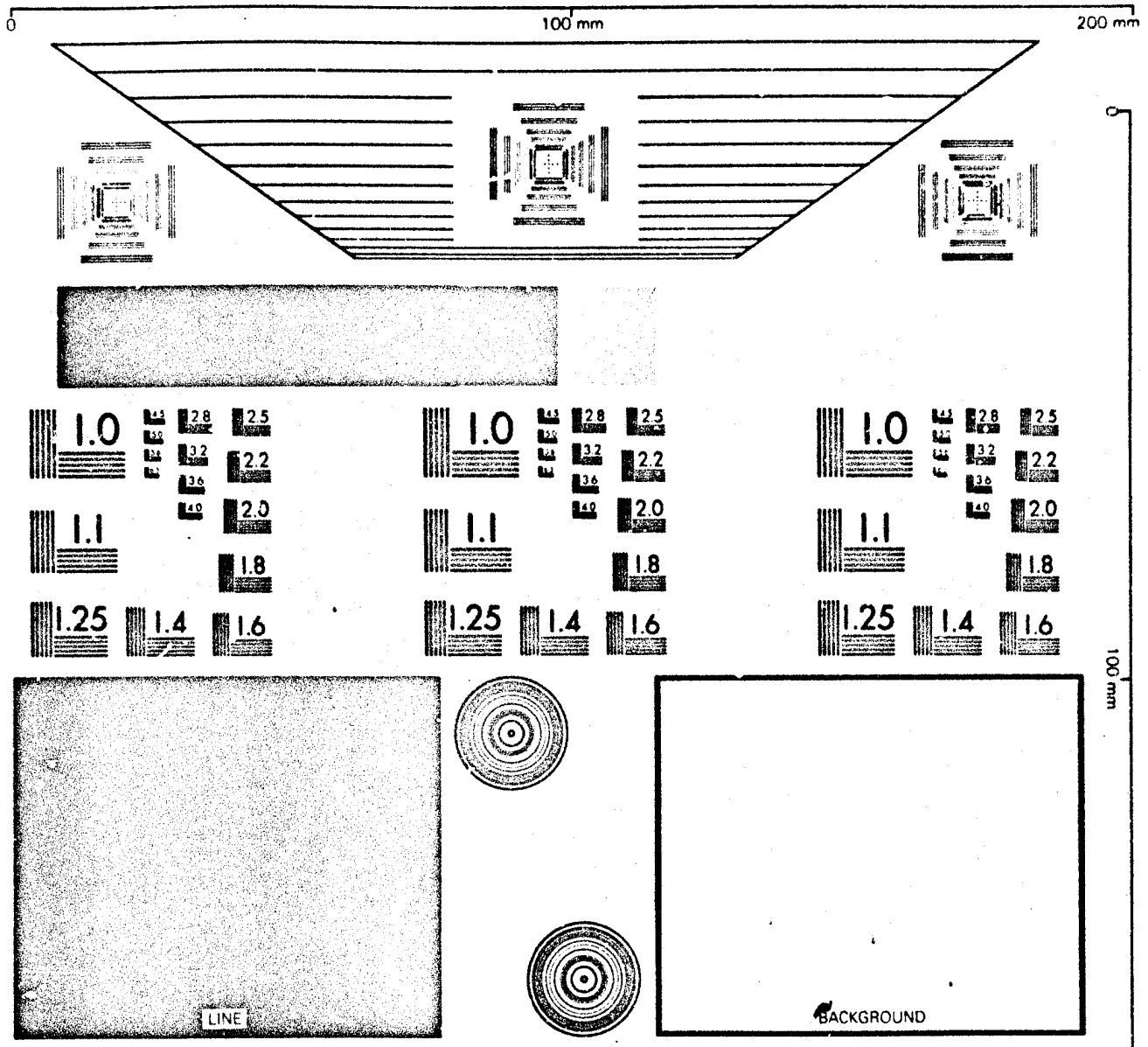
- *Test Activity Completion.* The user indicates if a test activity is completed, i.e., no more detected faults will be recorded.

A database server based relational database system, dBASE IV, is used in implementing the Data Collector, to take advantage of basic database read, write, and edit functions provided in the software package. A database software package will also be better equipped in the

CONTROL TEST TARGET

Kodak Quality Monitoring Program

2.4x



Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 5

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 1

Sensitized Imaging Products
· Business Imaging Systems Division

A-2289

7/87

management of a growing database. The menu-building facilities of dBASE IV have been used in user interface development.

Analysis of the ATLAS Code

In order to get the statement numbers of the Start-At (SAS), Start-At-Entry (SAE), and Halt-At (HAS) statements, it was necessary to examine the ATLAS source code for the transmitter diagnosis portion. Each test in the source code was examined, one at a time. The SAS, SAE, and HAS of each test in the program were recorded in a notebook. Also, the PCOF call-outs of each test were recorded. This process took about 3 weeks to do, and had to be repeated when a new version of the ATLAS code was released part way through the project.

The SAS, SAE, HAS, and PCOF information was entered into dBASE IV files for use by KATSS.

Installation and Field Test

The prototype software and its supporting equipment (a Sun Microsystems SPARCstation 1+ workstation) were delivered to the RTS Laboratory at CFB Cold Lake in October 1991. It has been in use there for the purpose of collecting data, with a planned test period of four months.

The principal objective of the field test is to determine whether the approach of heuristic resequencing based on statistical records will work when the program has access to real maintenance data. A secondary objective of the field test is to find out whether having KATSS as a separate auxiliary system will be satisfactory, or whether an integrated system is needed. Finally, we want to find out the opinions and suggestions of the RTS technicians with regards to improvements and other changes, especially with regard to the user interface.

Because of an unexpectedly low incidence of failure in AN/APG-65 radar transmitters, it is likely that the field study will be extended from the original 4 months to 10 months or more.

CONCLUSIONS

Although at the time this paper was written, the results of the field test were not yet available, we were able to reach some interim conclusions as a result of our analysis of the problem domain.

The first conclusion we reached was that this approach is not likely to yield a significant speed-up for this particular domain (radar transmitters). Part way through the project, the ATLAS code was re-written by Amtek Testware, and we were told that this resulted in a 40% reduction in the time needed to diagnose a transmitter. This factor alone greatly reduced any scope for improvement. (It also confirmed DND's initial belief that there was room for improvement in the efficiency of the RTS software.) In seeking a performance improvement in ATE software, the place to start is the improvement of existing conventional ATE code. Only when this has been done should the client investigate the use of more sophisticated technologies, such as AI.

The second conclusion was that this approach was impeded by the requirement to use SAEs when the resequencing algorithm reordered tests (that is, the general resequencing problem). Running the test from the SAE number instead of the SAS number means repeatedly running a lot of safety tests. The safety tests take a significant amount of time. The same safety tests might be re-run many times if a lot of resequencing takes place. The technicians told us that the tests would find nothing new each time they were run.

The third conclusion was that effective use of statistical techniques requires large volumes of data. It became apparent during the first three months of the field trial that it would take many months to

gather enough data for KATSS to be useful. An alternative technique that might have been more appropriate is case-based reasoning[6] [7].

The final conclusion is that using a system like KATSS in conjunction with an existing system like the RTS is inconvenient. Since there is no interconnection between the RTS and KATSS, the operators must enter information into KATSS manually. It would be much more convenient for the information to be entered automatically by the computer that is doing the diagnosis. Also, it would be better if the functionality of KATSS was just another module of an overall ATE system, rather than a separate system.

ACKNOWLEDGEMENTS

We would like to thank Major James Fera, Major Larry Glenesk, and Mr. Peter Shirtliff of DFTEM for the guidance and technical knowledge that they have supplied during this project. We also are grateful to Sgt. Webster, Sgt. Pitchen, and the other members of the RTS laboratory staff at CFB Cold Lake for doing the work of entering the required data into KATSS during the field trial. Finally, thanks to Mr. Mike Halasz of the National Research Council, for his ongoing involvement in the project.

REFERENCES

1. Hughes Aircraft Company. Built-In Test Data: FY'86 Radar Production (Document No. 18R-1064, Revision M). Hughes Aircraft Company, El Segundo, California, 1987.

---

[6] Late in the development stages of KATSS, we found out that another team at ARC had used case-based reasoning on a similar project for an industrial client, with good results. It had the advantage of being adaptable without requiring large quantities of data However, by that time it was too late to change our approach.

2. Hughes Aircraft Company. Radar Test Set Technical Manual: Intermediate Maintenance Principles of Operation. Hughes Aircraft Company, El Segundo, California, 1984.

3. Hughes Aircraft Company. Test Program Instruction. Revision E. Hughes Aircraft Company, El Segundo, California, 1987.

4. Author unspecified. Transmitter Functional End-to-End Test Flow Document. Internal DFTEM working document, 1990.

5. Allen, J. "Towards a General Theory of Action and Time." Artificial Intelligence, 23(2): 123-154, July 1984.

6. Aho, A. V., Hopcroft, J. E., and Ullman, J. D. The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, Massachusetts, 1974.

7. Riesbeck, C. and Schank, R. Inside Case-Based Reasoning. Lawrence Erlbaum Associates, Inc., Hillsdale, NJ, 1989.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

AN EXPERT SYSTEM APPLICATION FOR TROUBLESHOOTING THE CF18 F404 ENGINE

W.D.E. Allan[1] and R.C. Best[2]

Abstract

Troubleshooting the CF18 F404 engine often involves referring to data recorded by the Inflight Engine Condition Monitoring System (IECMS). The IECMS routinely records engine and aircraft parameters during flight. A software suite has been developed by GasTOPS Ltd. to graphically present this data. Troubleshooting procedures have been amended to include the analysis of the plots, focusing on features unique to known faults. This paper proposes an expert system application to provide assistance to technicians in utilizing this information in the troubleshooting process. The ability of the system to recognize fault signatures and to be readily updated with new patterns will be central to the success of the proposed application. The role of the system in supporting second and third line maintenance activities will also be discussed. An over-riding consideration in the application of expert system technology to the maintenance of the F404 is the necessity to maintain the level of expertise of the technicians. The system will provide recommendations for solutions; final maintenance decisions must always rest with the technician.

[1]Propulsion Engineer and [2]Propulsion Section Head, Mechanical Projects Branch, Aerospace Maintenance Development Unit, CFB Trenton

INTRODUCTION:

The commitment to on-condition maintenance for the CF18 aircraft has resulted in the development of numerous means of detecting component and system conditions. The maintenance of the F404 engine has been a major focus in these efforts. Hence, much of the groundwork is complete to support the development of the expert system proposed in this paper.

A vast amount of historical and contemporary CF18 fighter aircraft data is available supporting the F404 engine maintenance activities. Rapid access to maintenance information is key to preventing needless activity through the provision of timely, accurate and helpful troubleshooting. This paper constitutes a proposal for an expert system application for F404 troubleshooting, and comments on possibilities at other levels of maintenance.

DISCUSSION:

The two sources of data for F404 maintenance are the aircraft, and the maintainers at all levels. The aircraft is equipped with a Maintenance Status Data Recording System (MSDRS) which stores aircraft unserviceabilities as identified by the logic of the Mission Computers. Life cycle counts, operating exceedances, sensor failures and other anomalies related to the engines are recorded using a sub-system of the MSDRS, the Inflight Engine Condition Monitoring System (IECMS). This system also allows pilots to initiate records, and automatically records a thrust check during each takeoff. There is also a Maintenance Monitoring Panel (MMP) where numerical codes are set by aircraft logic identifying unserviceabilities or operating exceedances for ground crew action. All MSDRS data is stored on a magnetic tape which may be removed from the aircraft upon landing, and delivered to the Integrated Ground Data Station (IGDS) when required. The VAX-based IGDS system is capable of presenting data to the maintainers, and archiving the Aircraft Data Files (ADF) for future reference.

The second major source of maintenance data for the F404 is the maintainers themselves. Dispersed at three levels of maintenance, technicians rely on their own experience and judgement, in addition to standard procedures, to repair unserviceabilities or rebuild assemblies. The proposed system must be designed by these experts (technicians, maintenance managers and engineers) and supported by them. For the system

will rapidly become obsolete as an "expert" if it is not conscientiously and effectively modified to evolve with changing maintenance and information processing activities. The expert system application will be described in relation to the level of maintenance activity which it supports.

The First Level of Maintenance

The first level of maintenance involves the servicing of aircraft and the rectification of minor unserviceabilities. This includes any problem which can be solved in less than 24 hours.

The heart of automated first line activity is the First Line Troubleshooting System (FLTS) which consists of a laptop computer linked to the IGDS. The software is actually two of a suite of six distinct programs [1] which make up the Engine Performance Monitoring (EPM) system. These are: the Event Display Program, which graphically presents engine and flight parameter data acquired by the IECMS, and the Mission Analysis Program, which graphically presents low frequency data acquired throughout the flight for a limited number of engine and flight parameters. The latter is very useful in determining engine related faults when no MMP code, or pilot record is available for an aircrew reported problem. Its use still relies greatly on a technicians experience and technical expertise.

In using the Event Display Program, the first line technicians are provided with a Canadian Forces Technical Order (CFTO) which assists them in applying standard troubleshooting decision trees to the IECMS data. For a given engine-related unserviceability, the MMP code generated by the IECMS will provide the CFTO reference for identifying engine parameters of interest in the troubleshooting process. By visually analyzing the graphs and referring to the FLTS CFTO, the technician can usually isolate the fault, or at least, eliminate other possibilities. A sample of the Event Display program screen is shown in Figure 1.

The mechanism behind the proposed expert system at the first line of maintenance will be the determination of engine fault signatures in terms of the qualitative details of inflight events. The technicians rely heavily on the use of Canadian Forces Technical Orders to guide them in their work; decision trees and fault confirmation are logically laid out in these volumes. The proposed expert system will convert the

information held in these volumes into an on-screen, convenient and streamlined presentation of the decision criteria. (It will not be a step-by-step interactive activity.) As technicians work through the troubleshooting trees, characteristics of the graphical IECMS data will be analyzed to achieve a successful conclusion. Rectification recommendations would be available, for consideration by technicians in repairing the engine.

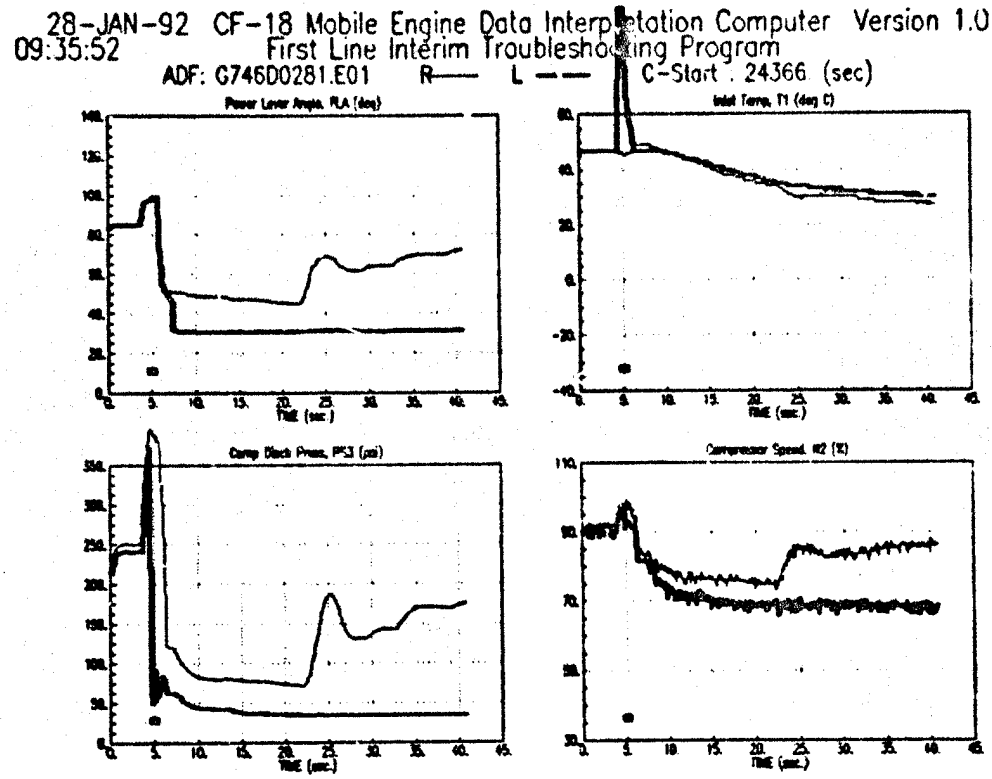


Figure 1: Event Plotting Program Screen Display Sample

A model of the F404 control system has been developed to assist the technician in analyzing graphs of IECMS data. Engine parameters and signals determined from control schedules can be displayed with actual engine data. This is intended to illuminate those qualities of the graphs which may indicate the source of problems. A dynamic model of the F404 engine has also been developed which will be capable of computer simulations of faults; this will likely prove invaluable in determining

more effective isolation methods for those rare, more troublesome faults.

In some cases, the MMP code history of the engine, or the maintenance history of a particular engine component, module or assembly may give the troubleshooter an indication of the possibility of a chronic problem. By accessing engine historical data, a first line technician could rapidly isolate a troublesome component or part.

Of paramount importance is the fact that the first line technicians must always maintain their technical expertise in troubleshooting. The proposed expert system must never be more than a ready source of expert or specific maintenance information. Decisions will always lie with the technicians and their supervisors.

The Second Level of Maintenance

The second level of maintenance carries out repair of major unserviceabilities on and off the aircraft, in-depth inspections, replacement of engine modules, and engine testing in the Engine Test Facility (ETF). In support of first line maintenance, the expert system will automatically compare IECMS data fault signatures with rectifications. This would also allow the second line technician to investigate the details surrounding cases where the actual local rectification differed from the recommendation given by the system. It is in this way that the fault library can be validated and improved. Problematic components or engines can be easily highlighted by the expert system, overcoming tedious manual searches, and case-by-case analyses.

The information from all F404 maintenance databases as shown in appendix 1 would be synthesized by an extension of the proposed expert system at second line. This would reveal a complete overview of F404 maintenance activities. Accurate forecasting of maintenance, overhaul, and spare parts use will be possible given the information available.

Figure 2 is a schematic diagram of the various components of the F404 maintenance information systems indicating how they will liaise with the proposed expert system.

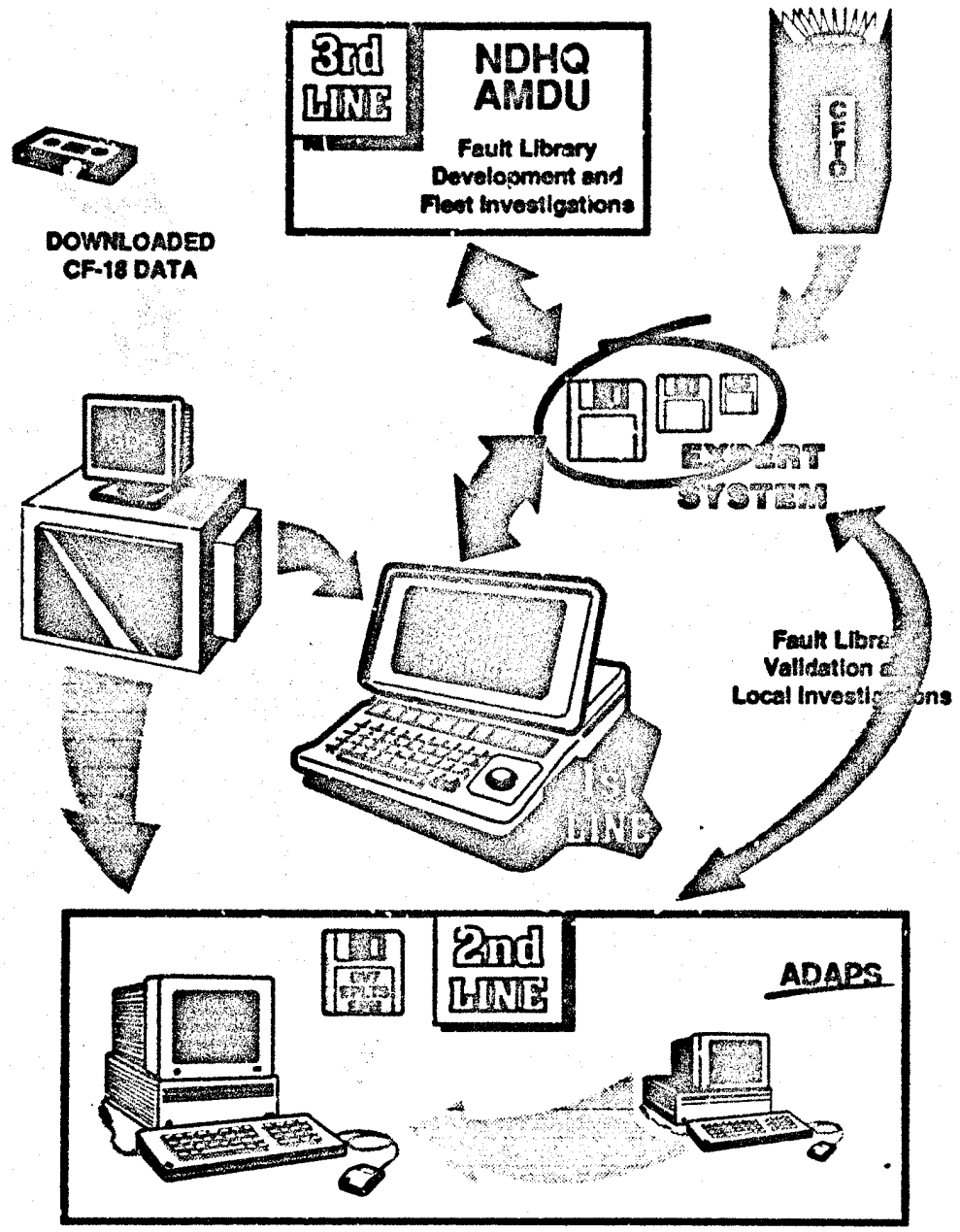


Figure 2 : The proposed Expert System for F404 troubleshooting

The Third Level of Maintenance and Development Establishments:

Development and enhancement of the proposed expert system will be initiated through engineering activities ongoing at third level agencies. For example, the system could, from a development point of view, provide complete and up to date fleet records to National Defense Headquarters and the Aerospace Maintenance Development Unit (AMDU). Studies and environmental comparisons could then be possible through ad hoc queries.

CONCLUSIONS:

An expert system has been proposed for the first line of F404 maintenance in the CF. First line technicians will use the system as a source of additional information for the purposes of troubleshooting the engines. The proposed expert system will highlight the unique characteristics of IEOMS records of unserviceabilities thus promoting timely and accurate diagnosis and repair. Second line technicians will find the nature of their tasks refocussed towards problem or suspect engines because the identification of these will be greatly facilitated by the expert system. Validation of the fault signatures will occur at the second line. Third line maintenance and development centres will support the development of the system by investigating F404 fleet data to develop and improve fault signatures.

Much stands to be gained by operators and maintainers alike, having a system which can promote rapid, reliable troubleshooting for a fleet of fighter engines. The proposed system would be a true advance for on-condition maintenance in the high-cost business of modern aircraft maintenance.

ACKNOWLEDGEMENTS

The Chief of Research and Development (CRAD) is currently funding the development of an expert system for F404 maintenance. This paper contains the Department of National Defense's preliminary concept of that system; greater detail will emerge as the project progresses. The discussions with Mr B.I. Forsyth of GasTOPS Ltd were extremely helpful in developing this work. The input of Mr J. Waring at the Director of Fighters and Trainers Engineering and Maintenance is also appreciated.

REFERENCES

1. Muir, D., File Note: "F404 Engine Performance Monitoring System Development", GasTOPS Ltd. File C-7-62.1, February 1990, pp. 2-5.

2. Military Support Department, Orenda Division, Hawker Siddeley Canada Inc, DV7 Reporting Guide, Orenda File 404-GEN-10-2, January 1992, 53 p.

3. Canadian Forces Technical Order C-14-404-000/AM-001, Engine Parts Life Tracking System - Policy Manual, Unclassified DND Document, pp 1-1 - 1-2.

The Aerospace Maintenance Development Unit

A lodger unit at Canadian Forces Base Trenton, the AMDU is the CF centre of expertise in Engine Performance Monitoring. Projects include on-wing performance trending and engine test facility instrumentation monitoring. The technical assistance provided in the development and field testing of the First Line Troubleshooting System was part of AMDU Projects D4151E, D0015E, and I1135E.

APPENDIX 1
F404 MAINTENANCE DATA BASES

Generic:

1. AMMIS: Aircraft Maintenance Management Information System: the general data base used throughout the Air Force for recording maintenance information.

F404 Maintenance Specific

1. DV7 A maintenance action data base which documents field experience [2]

2. EPLTS Engine Parts Life Tracking System provides a means of tracking Life Usage Indices (LUIs) for engine maintenance scheduling [3]

3. EPM Engine Performance Monitoring System relies primarily on data provided by the IECMS and the Automatic Data Acquisition and Processing System (ADAPS) in the ETF. The following six programs make up this suite of software [1]:

   a) The Event Plotting program graphically plots the IECMS data for the first line technician to refer to when troubleshooting;

   b) The Mission Analysis program graphically presents low frequency data acquired throughout the flight for a limited number of engine and flight parameters;

   c) The MMP Code program tracks MMP codes by aircraft tail number;

   d) The Take-off Analysis program processes dynamic engine performance data gathered in take-off thrust checks to determine engine health indices (EHIs) warning of performance degradation;

   e) The Take-off Trending program trends these EHIs; and,

   f) ETF Troubleshooting program, which assists in troubleshooting engines by matching performance parameters measured in the ETF to baseline values. This reveals a characteristic signature which can be compared to known fault signatures.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS II, DND

RMC, Kingston, Ontario, Canada

A DIAGNOSTIC EXPERT SYSTEM FOR DIGITAL CIRCUITS

Capt R.W. Backlund[1] and Dr J.D. Wilson[2]

Abstract

This paper presents a scheme for a diagnostic expert system which is capable of trouble-shooting a faulty digital circuit or producing a reduced test vector set for a non-faulty digital circuit. It is based on practical fault-finding logic and utilizes AI techniques. The program uses expert knowledge comprised of two components: that which is contained within the program in the form of rules and heuristics, and that which is derived from the circuit under test in the form of specific device information. Using both forward and backward tracking algorithms, signal paths comprised of device and gate interconnections are identified from each output pin to the primary input pins which have effect on them. Beginning at the output, the program proceeds to validate each device in each signal path by forward propagating test values through the device to the output, and backward propagating the same values to the primary inputs. All devices in the circuit are monitored for each test applied and their performance is recorded. Device or gate validation occurs when the recorded history shows that a device has been toggled successfully through all necessary states. When run on a circuit which does not contain a fault, the program determines a reduced test vector set for that circuit.

[1]Graduate Student, [2]Professor, Dept of Electrical and Computer Engineering, RMC, Kingston

INTRODUCTION

There was a time when fault-finding electronic equipment was a straightforward process which technicians could accomplish using logical and methodical testing procedures. This was, of course, back when circuits were almost totally accessible to intrusive testing techniques throughout their signal paths. Within the last two decades Printed Circuit Boards (*PCBs*) have become extremely complex in both design and operation, so that today they are typically several layers deep and capable of performing a multitude of consecutive and sequential oporations. Not surprisingly, circuit accessibility has decreased dramatically, to the point where many modern circuits are accessible at few points between their inputs and outputs. As a result, these modern circuits have become extremely difficult, and sometimes impossible, to diagnose using conventional technician-oriented methods. The general trend in both industry and the Canadian Forces has been to rely less on the technician and more on specialized Automated Test Equipment (*ATE*) for PCB diagnostics. This change in tactics has been precipitated by the fact that technician-based fault-finding requires considerable operating resources when compared to PCB-specific ATE systems, especially in the areas of training, tools and time required to diagnose faults.

ATEs operate in a systematic fashion, firing through a predetermined pattern of tests developed by domain experts, all the while monitoring the output responses of the circuit under test. The patterns of tests are referred to as *test vectors* and are elements of the comprehensive *test vector set* for the circuit which ensures that all devices and gates are verified according to the domain expert's specifications. Unfortunately, ATEs are usually expensive, as there are large costs involved in the design and development stages. One reason for this is that a considerable amount of design time is spent generating the test vector set, since it is important that the set be correct and complete. As can be envisioned, as the circuit becomes more complex the number of test vectors required also becomes larger. Since the number of possible test vectors for any given circuit is 2^N for N primary inputs, the test vector set grows exponentially rather than linearly for the number of inputs to the circuit. Schemes which can generate reduced test vector sets are therefore required in order to limit the time and resources spent in test pattern design as well as in the diagnosis of the circuits later.

Another shortcoming of ATEs is that substantial costs can be incurred later in their operational lives when system modifications are required to accommodate changes in hardware and/or software of unit(s) they test. This situation occurs since the test vector sets are circuit-specific, and apply

uniquely to one type of circuit. Artificial Intelligence (AI) techniques offer a possible solution to this problem by introducing intelligence, through the utilization of rules and heuristics, to the program's generation and application of test vectors. Such rules and heuristics can ensure that test vectors are applied in a logical, as opposed to random or sequential, fashion. It can thus be possible to fault-find a given circuit using significantly fewer test vectors than contained in the comprehensive test vector set. AI techniques can also ameliorate the reduced test vector generation process, obviously a critical part of circuit design.

Two basic strategies exist with respect to test vector generation: statistical, or pseudo-random, test vector generation and algorithmic test vector generation. In the statistical method, a circuit is bombarded by random or heuristically chosen test patterns which are added to the test vector set if and only if they enable the detection of a previously undetected fault condition. With this method, those faults which are easily detected are located fairly rapidly, while those which are more difficult to detect require increasing computation time. An upper limit may have to be set on computation time or the number of test vectors comprising the test vector set as full fault coverage may require excessive computation time. In the algorithmic method, a specific algorithm is used to generate a set of test vectors for each identified fault in the circuit. Most such algorithms also require considerable computation time and may contain redundant tests, as no record is maintained of other device performance during a specific test on a specific device or gate. Additionally, neither of the above-mentioned methods attempt to handle sequential devices such as flip-flops, latches and registers, and therefore are of little use in diagnosing computer interface devices or mode command latches.

Work at RMC in the general area of automated fault-finding was begun in 1989 by Capt Ron Boyce[1]. In his thesis, presented in 1990, he introduced a FORTH language-based PCB diagnostic expert system for use with the CP. This work was extended by Capt R.E. Leroux[2], when in May 1991 he presented an IBM PC/AT-based Data-Driven Expert System (DDES) for PCB diagnostics. This system utilized the ORACLE RDBMS for storage of expert 'knowledge', and used a C-based 'knowledge processor' to manipulate the database. A plotter table incorporating a mechanical probe permitted automated intrusive point-testing anywhere on the surface of the PCB, with test data acquired from either the PCB's edge connector or from the mechanical probe. Test data was processed by a Burr-Brown data acquisition board, which also controlled test signals to the PCB. Since the 'expert knowledge' was held in a database, changes to this information could be made without re-compiling the entire system, unless changes were made to the data-input

forms (written in SQL*FORMS). The system could be configured for either diagnostic or teaching purposes, and was demonstrated on a PCB which incorporated both analog and digital signal processing circuitry.

THE DYNASTIC EXPERT SYSTEM

The DYNASTIC (dynamic diagnostic) expert system presented in this paper uses the AI techniques of forward and backward chaining, and contains expert knowledge embedded within its rules and heuristics. It is capable of trouble-shooting a digital circuit in a logical manner, using heuristics and rules similar to those which a human trouble-shooter might use. Although it is algorithmic in nature, the DYNASTIC Expert System is capable of both fault-finding and reduced test vector set generation. When the program is run on a faulty circuit, it generates sufficient test vectors to permit isolation of a fault by validating all appropriate circuit devices, from output backwards, in the faulty signal path. Once the fault is located, the program indicates the device which must be replaced, and then stops. If more than one fault exists, the program can be run again, once the faulty devices which have been identified have been replaced. Since the DYNASTIC Expert System generates only enough test vectors to isolate a fault, when it is run on a circuit which does not contain a fault, it will continue its search through all paths in the circuit, ultimately validating all devices in the circuit and generating a reduced test vector set as it does so.

To avoid generating duplicate test vectors, the DYNASTIC Expert System maintains a history of node values for all devices in the circuit at all times. The program will test a device with a test pattern only if that particular pattern has not been applied before, either as a direct or indirect result of testing performed on the circuit. The program provides test-by-test feedback to the user in the form of all node output values, the device being tested, the input pattern being applied to the device under test, and device validation as it occurs. This information is currently provided in scrolling text format, however a fully developed system could present this information on a graphical display of the circuit under test.

The DYNASTIC Expert System uses 5-valued logic which consists of two representation for logic 0, *F* and *f*, two representations for logic 1, *T* and *t*, and a Don't Care value of x. A *STATIC* 'F' or 'T' representation is used when there is no choice as to the value which must be applied at an input pin of a device, and a *NON-STATIC* 'f' or 't' representation is used when there is a choice. *STATIC* values originate as components of the main test pattern which is being applied to a particular device to validate it, and are then propagated backwards to the primary inputs. As an example, in "Fig

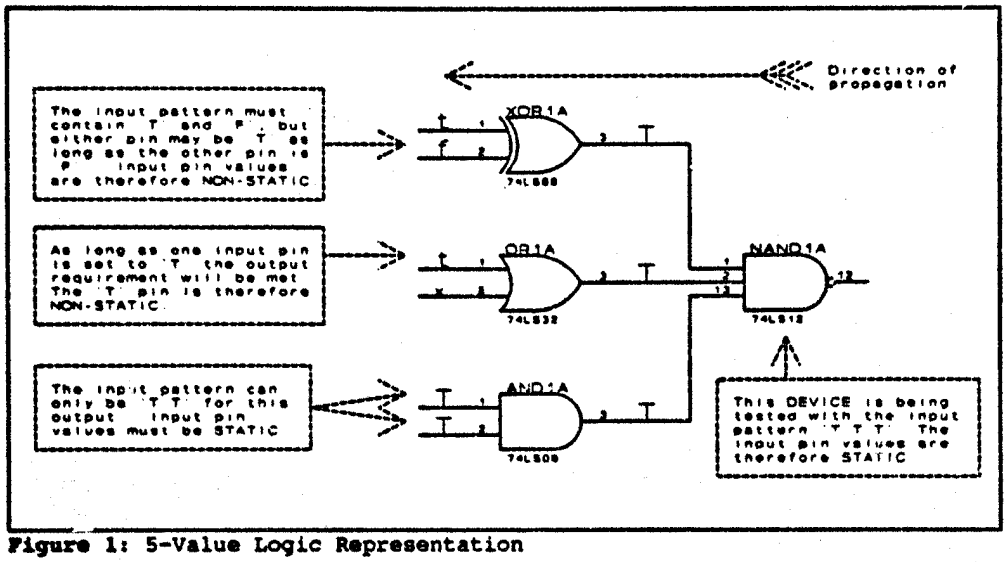


Figure 1: 5-Value Logic Representation

1" NAND1A is being tested with the pattern *'T T T'*. The values which comprise this test pattern are *STATIC* since they cannot be changed without changing the very nature of the test. The static 'T' value on pin 1 of NAND1A is backward-propagated through pin 3 of XOR1A, where an input pattern of either *'t f'* or *'f t'* will satisfy the output requirement. Since there is more than one way to achieve the required gate output, the input values are *NON-STATIC*. The static 'T' value on pin 2 of NAND1A is backward-propagated through pin 3 of OR1A. To satisfy its output requirement, OR1A's input pattern can be either *'t x'* or *'x t'*, where x can be either 't' or 'f'. Again, since there is more than one way to achieve the required gate output, the 't' input value is *NON-STATIC* as is, of course, the 'x' value. The static 'T' value on pin 13 of NAND1A is backward-propagated through pin 3 of AND1A, where the only possible input pattern which will produce the required gate output is *'T T'*. Both of AND1A's input values are therefore *STATIC*. The STATIC and NON-STATIC values on the input pins of XOR1A, OR1A and AND1A are backward-propagated in a similar fashion through other devices in the signal path until the primary input pins are reached.

Circuit information must be supplied to the DYNASTIC Expert System in the form of a specially formatted netlist. At present, this netlist is supplied through ORCAD, which is also the circuit development environment. The netlist can, however be manually produced on a standard word-processor, or alternately could be supplied by the manufacturer of a particular PCB or device. The DYNASTIC Expert System analyzes the netlist information for the circuit, and from it identifies all primary inputs and outputs and pin

designations. It then uses a back-tracking algorithm to determine the signal paths from each output pin back to the primary input pins which have effect on it. Device inter-connections are logged as the program proceeds along each path. Once the signal paths have been identified, the program chooses the signal path containing the largest number of primary inputs and the largest number of gates and begins to diagnose it. By choosing the longest path first, the program effectively tests the maximum number of devices at the same time, since alternate test patterns will concurrently appear on the input pins of other devices while any single device is being validated. These particular test patterns will not have to be applied again when it is time to validate these other devices.

Working from the output backwards, each device or gate in the signal path is in turn validated by applying on of the required test patterns for that device if, as mentioned above, that particular test pattern has not appeared on the device's input pins before as an indirect consequence of a test on another device. The test patterns are backward-propagated to the appropriate primary input pins, and forward propagated to the output. As shown in "Fig 2a", at some point in the backtracking process, it is possible that a particular input pattern for a device will require input patterns in devices before it in the signal path which cannot possibly occur concurrently. If such a contention of signals occurs, a conflict resolution

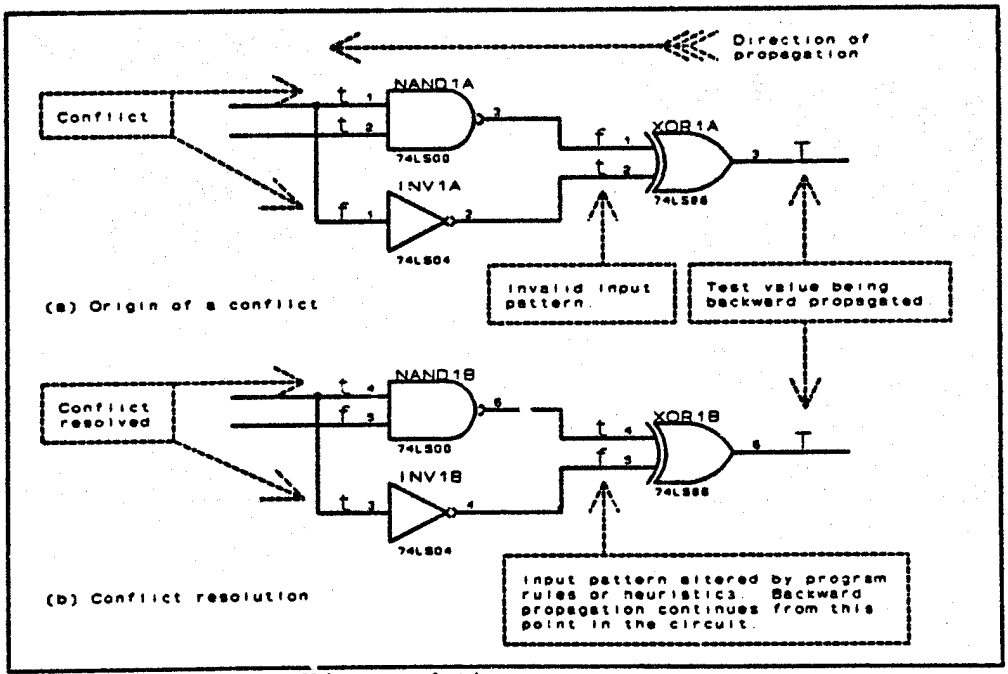


Figure 2: Example conflict resolution process

algorithm is applied to the circuit which back-traces to the source of the conflict and applies rules and/or heuristics to resolve it. The algorithm then indicates to the main program where to resume backward-propagation, as indicated in "Fig 2b". If a test cannot possibly be conducted due to the physical interconnections of the circuit, the DYNASTIC Expert System identifies the particular device which could not be fully validated, and indicates the invalid test pattern. In "Fig 3", a conflict is created at the inter-connected input pins of NOR1A and NAND1A by the required test pattern 'T F' being backward propagated from the input pins of DEVICE X. Since this conflict is unresolvable, the test pattern 'T F' is considered *invalid* for DEVICE X. Fortunately, some of the 2^n test patterns are

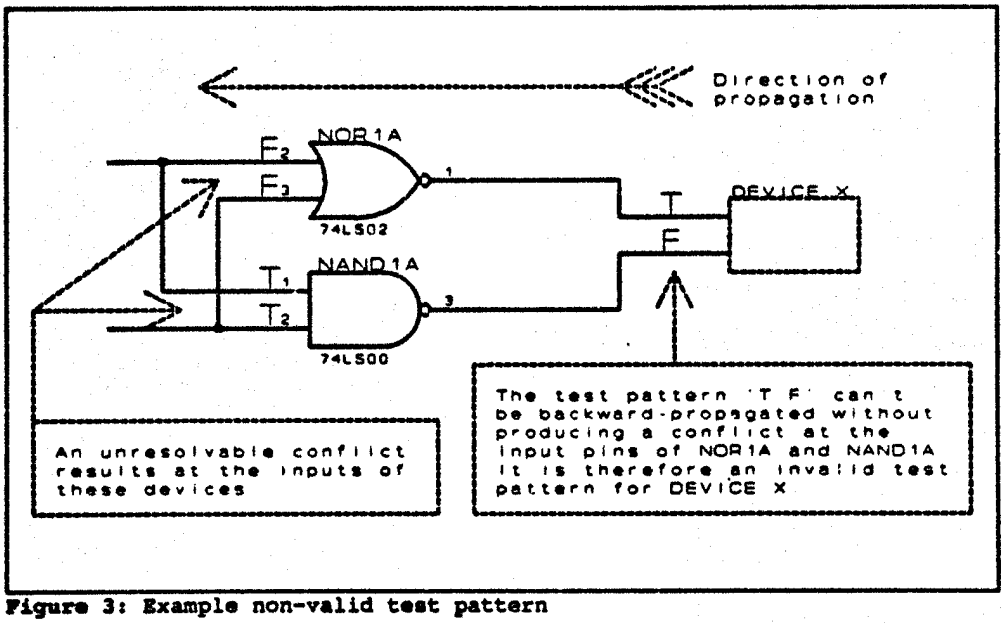


Figure 3: Example non-valid test pattern

redundant for many devices, and so a device can still be validated even though all possible input combinations are not exercised. To more fully understand this concept, a look at the internal construction of a logic device is required. "Fig 4" shows a 2-input CMOS AND gate. Internal construction for TTL and other types of AND gates is similar with the exception of the typn of transistors utilized and the types of input and output buffering. As can be seen in "Fig 4", if either Input 1 or Input 2 are low, transistor Q2 will be cut-off and transistor Q5 will be turned-on. Inputs 1 and 2 therefore need not be low at the same time, as this will offer no new information about the circuit. The 2-input CMOS AND gate thus requires the test patterns '*T F*', '*F T*' and '*T T*' to validate it, and test

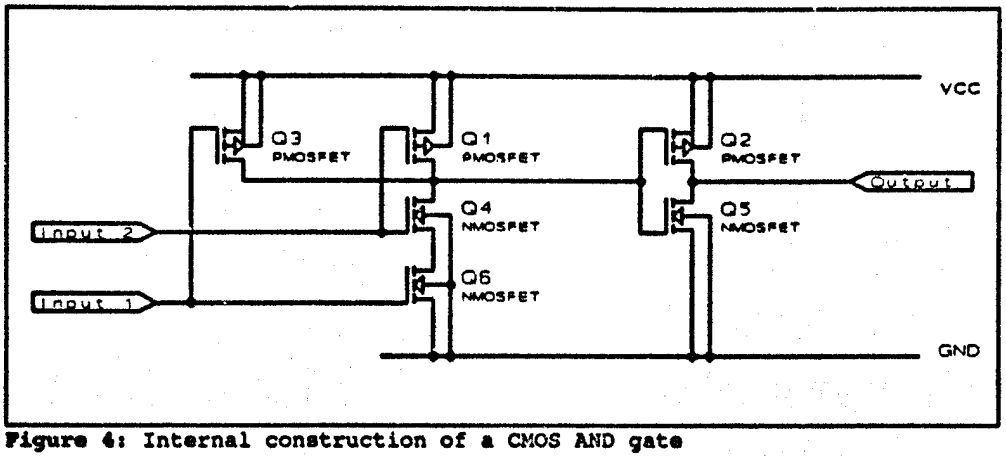


Figure 4: Internal construction of a CMOS AND gate

pattern '*F F*' is superfluous. Likewise for a 3-input CMOS AND gate, the test patterns '*T T T*', '*F F T*', '*F T F*', and '*T F F*' are absolutely required for device validation, and the rest of the 2^n possible input combinations are superfluous. This principle applies to all AND gates regardless of the number of inputs, with *n+1* (where n=number of inputs) test patterns being absolutely required for device validation. This type of test pattern reduction can also be applied to logic NAND, OR, and NOR gates, with *n+1* test patterns required for validation of these devices. The DYNASTIC Expert System generates device test patterns based on this knowledge of device behaviour.

Once all conflicts have been resolved and all logic values have been determined for a particular path, the circuit primary inputs are forward-propagated to the circuit outputs. They are then verified as either correct or incorrect by comparing them to the physical circuit's actual outputs. As mentioned above, if the circuit does not contain any faults, the program will generate a set of test vectors which will validate all devices and which will be non-redundant. This set of test vectors can then be stored in a library of such test vectors and used for analysis of similar circuits when they occur as sub-sets of larger circuits.

If at any time during testing one or more of the outputs of the physical circuit do not agree with the DYNASTIC Expert System-generated values, the program goes into a fault isolation mode and proceeds to isolate and trouble-shoot the indicated faulty path from output to input. While in this fault-isolation mode, if a faulty output value is indicated the program attempts to further isolate the source of the fault by seeking alternate routes within the path through which it can verify the current device-under-test. Additionally, alternate signal paths (feeding different outputs)

which contain the current device-under-test are investigated, with the current test pattern being propagated through to these different outputs. If these tests fail, the current device-under-test must be considered to be possibly responsible for the fault condition. All devices connected to its inputs are then validated, if possible, using the above mentioned fault-isolation procedures. As many devices as possible are validated before the program finally produces a list of the device(s) which may be responsible for the particular fault condition.

For an initial test of the DYNASTIC Expert System, the 4-bit full-adder circuit of "Fig 5" was selected. This circuit is similar in construction to the 74LS83/74LS283 chip, and so presented an easily verified, non-trivial test environment. A full mock-up using LS TTL components was used to

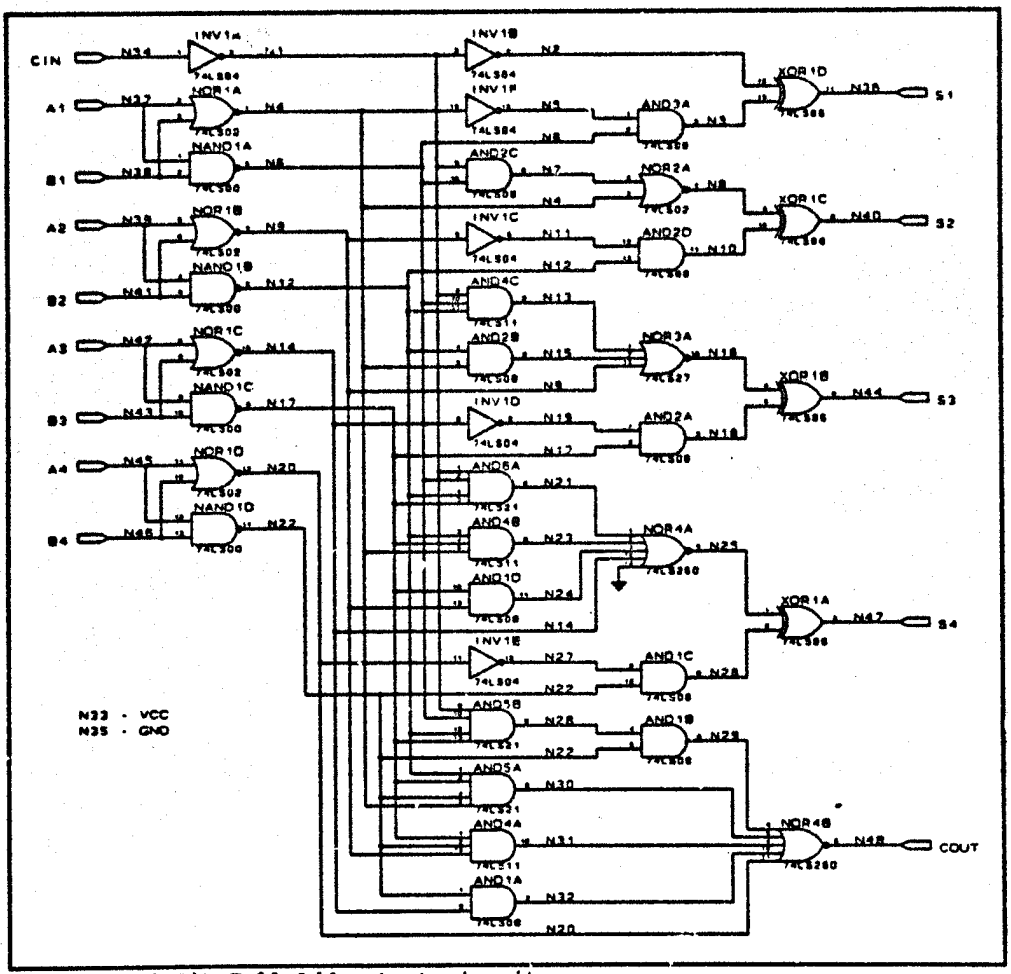


Figure 5: 4-Bit Full Adder test circuit

simulate faults, with an actual 74LS83 chip used as a reference. The comprehensive test vector set for this circuit is 2^9 or 512. The DYNASTIC Expert System determined that a total of 7 test vectors were required to fully test all devices in this circuit (see "Table 1"). Of the 7 vectors generated, none were redundant, and all gates in the circuit were validated.

	PRIMARY INPUTS									OUTPUTS				
TEST VECTOR APPLIED	C IN	A 1	B 1	A 2	B 2	A 3	B 3	A 4	B 4	S 1	S 2	S 3	S 4	C OUT
1	F	F	T	F	T	F	T	F	T	T	T	T	T	F
2	T	F	F	T	F	T	F	T	F	T	T	T	T	F
3	T	T	F	F	F	F	T	F	T	F	T	T	T	F
4	F	F	F	T	T	F	F	F	T	F	F	T	T	F
5	F	T	T	F	T	F	T	F	F	F	F	F	T	F
6	F	F	F	F	F	F	F	T	T	F	F	F	F	T
7	F	F	F	F	F	T	T	F	T	F	F	F	F	T

Table 1: Reduced Test Vector Set for 4-Bit Full-Adder circuit

Fault simulation has only recently been attempted on a limited scale, with encouraging results. As an example, a stuck-at-0 fault was simulated on N7 of the 4-bit full-adder circuit. The DYNASTIC program generated 17 test vectors before isolating the fault to either of devices AND2C or NOR2A, both possible culprits for this particular fault. A stuck-at-1 fault simulated on N19 required 17 test vectors to isolate it to either of devices AND2A or INV1D, again, both possible culprits for this particular fault. In both examples above, the DYNASTIC Expert System verified that there were no other devices involved in the particular fault by validating them through alternate signal routes and signal paths.

FUTURE WORK

Work is presently underway which will give the DYNASTIC Expert System the capability to fault-find sequential devices, such as flip-flops, latches and shift registers. An improved user interface is planned which will include mouse, icon and keyboard input, and a graphic display of the circuit being tested. The DYNASTIC Expert System's progress will then be graphically displayed on-screen as the circuit is processed. An improved fault-location algorithm is being developed which will allow faster identification of a fault condition. A library system containing test vector sets for commonly used sub-circuits is planned. This library system will allow the program to treat such sub-circuits as self-contained devices which can then be quickly validated with the pre-determined test vector sets. The test vector generation and fault-finding processes for large and

complex circuits should thus be speeded up considerably. It is also anticipated that automatic data acquisition and signal generation can be incorporated into the DYNASTIC Expert System, so that a circuit can be connected to the program and diagnosed automatically without user intervention. This mode of operation will be in addition to the present manual mode, which can then be configured for instructional purposes.

CONCLUSIONS

The DYNASTIC Expert System was developed with the intent of emulating practical trouble-shooting techniques. To this end, it has been successful. As an indirect result of the implementation of this methodology, it has been shown that the program is also capable of generating reduced test vector sets for the circuits analyzed. The program is still under development and so has limitations to the types and sizes of circuits it can diagnose, however, the expert-based methodology implemented should prove to be appropriate to almost any type of digital circuit.

REFERENCES

1. Boyce, Ronald, "Object-Oriented Diagnostics for Printed Circuit Boards", Master of Computer Engineering Thesis, RMC, Kingston, Ontario, Canada, 1990.

2. Leroux, Robert, "Data-Driven Expert System for PC Board Diagnostics", Master of Computer Engineering Thesis, RMC, Kingston, Ontario, Canada, 1991.

3. Klenke, R.H., Williams, R.D. and Aylor, J.H., "Parallel-Processing Techniques for Automatic Test Pattern Generation", Computer Magazine, Vol. 25, No. 1, IEEE Computer Society, January 1992, pp. 71-84.

4. Novellino, J., "Test-Vector Generator Handles Sequential Logic", Electronic Design, Vol. 38, No. 9, May 1990, pp. 119-120.

5. Raina, R. and Rajashekhara, T.N., "A Generalized Fault Simulator for Combinational Logic Circuits", Computers & Electrical Engineering, Vol. 15, No. 3-4, 1989, pp. 89-96.

6. Vishiwani, A.D., Kwang-Ting, C. and Prathima, A., "A Directed Search Method for Test Generation Using a Concurrent Simulator", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 8, No. 2, 1989, pp. 131-138.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

MANAGEMENT OF UNCERTAINTY IN KNOWLEDGE-BASED SYSTEMS: A MILITARY PERSPECTIVE

Maj M.C. McKean,[1] Dr. J. Fugère,[2] and Dr. L. Niem[3]

Abstract

Today's military commander is faced with a complex and uncertain world. Artificial intelligence (AI) systems, specifically knowledge-based or expert systems, are currently being developed as a means of providing intelligent assistance to a variety of military problems. From the perspective of the Defense Advanced Research Projects Agency of the US Department of Defense (DARPA) "... we stand at the threshold of a new generation of computer technology having unprecedented capabilities". However, as evidenced by practical experience, the realization of these capabilities is contingent on the management of uncertainty.

This paper surveys the principal approaches used to manage uncertainty, in knowledge-based systems and examines their implications from a military perspective.

[1]Graduate Student, [2]Associate Professor, and [3]Assistant Professor, Dept of Mathematics and Computer Science, RMC, Kingston

2

Introduction

That the world is a complex and changing place, fraught with uncertainty and risk, should come as no surprise to anyone. The military professional is well aware that perceptions can be incorrect, and that it is seldom possible to know what tomorrow will bring (e.g., the recent history of the Soviet Union). However, it is equally clear that selecting a course of action in an effort to solve a given problem (e.g., dealing with what Mr. Baker refers to as the "loose nuke" problem), is influenced by available knowledge, and that mistakes in the processing of this knowledge are often attributable to human errors [2-4].

The advent of the computer, during the 20th century, raised hopes that an intelligent machine could be developed that would overcome these human limitations, and would be capable of providing the answers to many (if not all) of the problems faced by mankind. Research aimed at producing such an intelligent machine, fuelled by dreams of space travel and the desire to build ever more efficient weapons of war, has focused on AI work related to knowledge-based systems. This research, which has grown exponentially during the last decade [5, 6], has resulted in a number of important advances. However, important design issues remain, and "uncertainty management is a case in point [7, p. 29]".

Few questions have led to so much argument and so little enlightenment as the question "What works?" in the management of uncertainty. A review of the literature reveals the existence of a multitude of approaches for representing and reasoning with uncertain information [7-12], and the absence of an accepted methodology for implementing this technology [13-21]. This paper provides an overview of the uncertainty management problem, and examines the implications of this technology from a military perspective [22-31], by reviewing how Bayesian belief networks can be applied to a military classification problem [32-38].

Uncertainty

Uncertainty, which is an inexact concept, can be affected by information that is: uncertain, incomplete, or inconsistent [2, 3]; or by heuristics which inaccurately describe relationships between different pieces of information [11]. Both the reasoning process and knowledge can be affected by uncertainty, and errors that occur when information and/or heuristics are combined in an effort to discover new knowledge (such as the process of problem solving [13, 14]) can complicate this situation. Uncertainty is typically explained through the use of informal explanations and examples [11], and this paper examines the question of uncertainty through the eyes of a soldier who must decide whether to attack or welcome an unidentified agent (Figure 1).

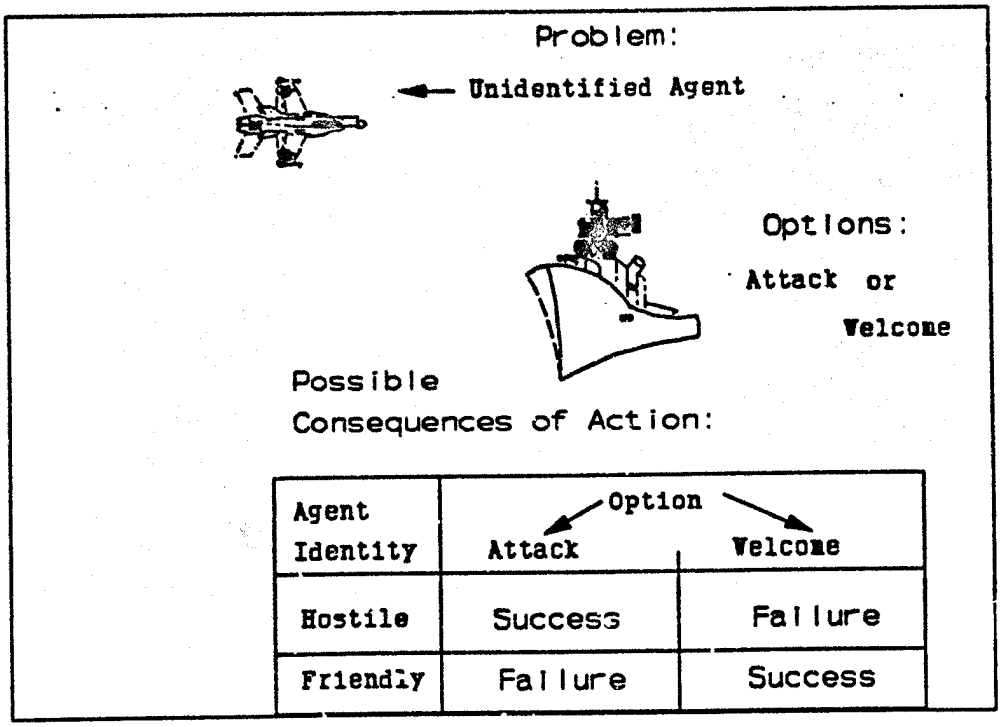


Figure 1. The identification Friend or Foe (IFF) Problem

Knowledge-Based Systems

Knowledge-based systems are computer systems that consist of a knowledge base, and a knowledge base manager [16]. Although this technology is still evolving and no standard architecture has yet emerged [15], knowledge-based systems typically operate in a cooperative environment (Figure 2).

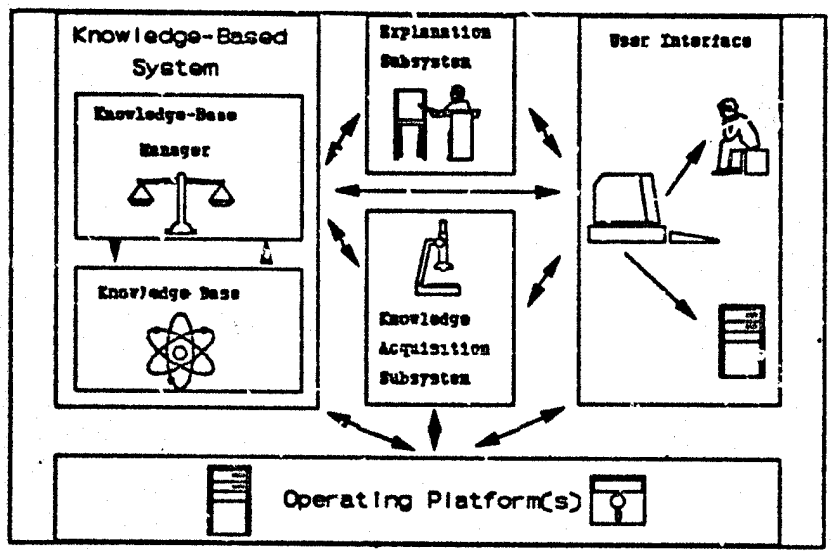


Figure 2. The Knowledge-Based System Environment

The development of knowledge-based technology can be traced to a number of events during the late 1970s and early 1980s, many of which were motivated by military interest in intelligent reasoning systems [10]. This research led to the fielding of a variety systems which were widely embraced in the belief that they would lead to tremendous economic payoff [15]. Although this technology has led to dramatic changes throughout the industrialized world [17-20], these systems lack a number of important capabilities [10]. Efforts aimed at overcoming these limitations have focused on the management of uncertainty, and have led to the development of a multitude of approaches for the management of uncertainty [11]. What, then, are these uncertainty management approaches?

Uncertainty Management Approaches

Uncertainty management approaches, which span the spectrum from ad hoc programming solutions to formal representational languages complete with mathematical proofs, encompass a multitude of techniques designed to deal with uncertain, incomplete, and inconsistent information [11]. Although an examination of all of these approaches is clearly beyond the scope of this paper (Note 3), the rationale for selecting a specific formalism can be articulated according to: the type and structure of information to be processed, task type, and the development framework. Each of these issues is discussed briefly prior to examining what is done in practice, and working through an example using Bayesian belief networks.

Type and Structure of Information

The type and structure of information that is available to be processed by knowledge-based systems can broadly be classified as either: numeric (quantitative), or non-numeric (qualitative) information. For this reason a number of different uncertainty management approaches have been developed, and can be classified as either: numeric or non-numeric methods, according to the type and structure of the information that they are designed to process (Table 1).

Table 1. Uncertainty Management Formalisms

Numeric Methods	Non-Numeric Methods
One-Valued Quantitative Formalisms Baye's Theory, Belief Networks...	Multivalued Logics Modal Logic, Default Logic,...
Two-Valued Quantitative Formalisms Dempster-Shafer Theory...	Heuristic Techniques Endorsements...
Set-Valued Quantitative Formalisms Fuzzy Set Theory...	Biological Processes Neural Networks

Although much of the uncertainty literature appears to be preoccupied with a debate over which uncertainty management approach is the "best", there is a growing body of evidence which argues that a more open-minded approach is required [8, 11]. This integrative approach, which is based on the contention that different types of uncertainty can most efficiently be managed by employing a framework of different uncertainty management approaches, is closely associated with research aimed at decomposing application problem domains according to generic task types.

Knowledge-Based System Task Types

Knowledge-based systems have been applied to a number of different problem domains (DIALOG employs 18 subject descriptors [6]); however, an analysis of these applications reveals the existence of a number of generic task types (Figure 3). This approach is based on the belief that there is no one "best" method of managing uncertainty, that different problem types involve various types of uncertainty, and that the determination of an appropriate uncertainty management approach is dependent on the type of task [8].

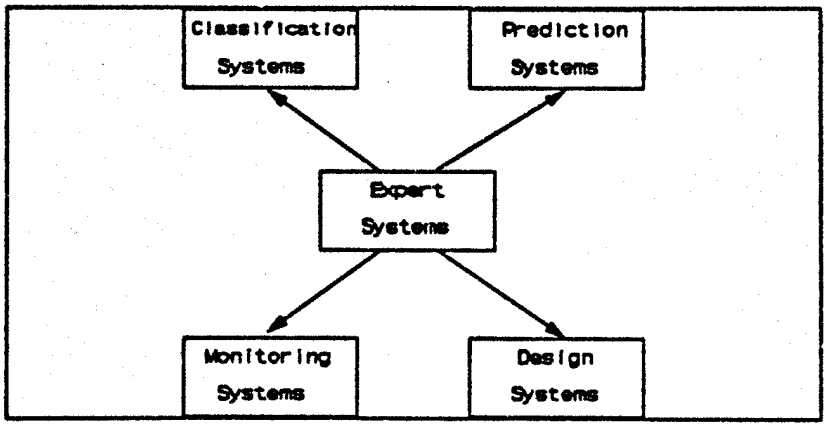


Figure 3. Taxonomy of Expert Systems - By Type of Task [9, p. 10]

Frameworks for Developing Knowledge-Based Systems

The development of knowledge-based systems, capable of supporting the complex requirements of dynamic large scale problems, has led to the adoption of new methodologies, architectures, and system development tools [10, 11]. Although standards are still evolving, these next-generation frameworks typically permit the selection of appropriate uncertainty management approaches in accordance with task type requirements, and emphasize modular construction. Where reusable component parts designed to efficiently solve specific tasks (e.g., the IFF problem), can communicate to cooperatively solve problems (Figure 4).

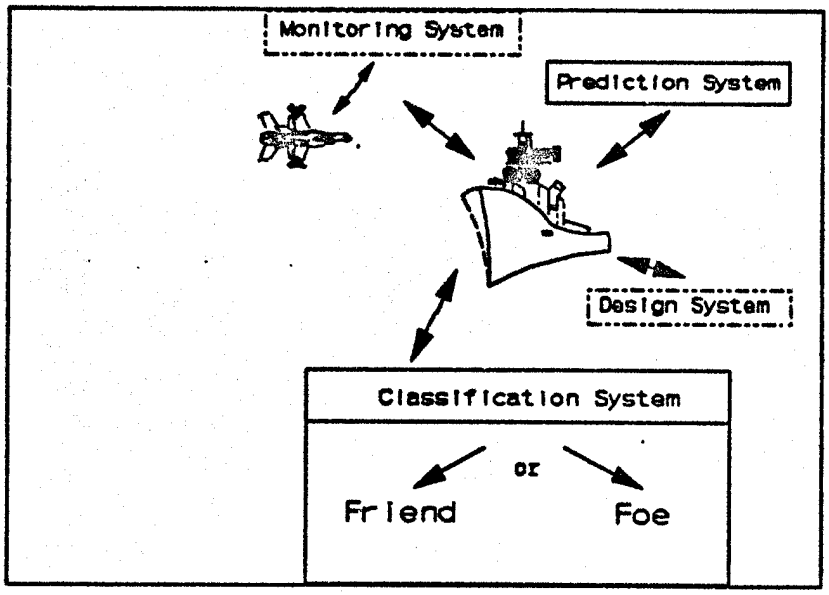


Figure 4. Cooperative Frameworks for Knowledge-Based Systems

Although the rationale for adopting a cooperative framework can be demonstrated at a conceptual level, the most convincing arguments for a cooperative approach come from a comparative examination of single and multiple computational models [11]. This is accomplished, in the following section, by examining the implemented system literature.

Implemented Systems

Implemented systems, which can appear in many different shapes and sizes, have been defined as computer based systems that are capable of solving or assisting in the solution of practical problems [11]. Although a cursory review of the literature could lead one to believe that only a handful of implemented knowledge-based systems exist [21], numerous applications can be found [6]. These implemented systems have had a dramatic effect on manufacturing processes [17-20], and there is a general sense that this technology, which is still evolving [10], has been accepted as a must have in the commercial sector [15]. What, then, is this technology, and what capabilities does it possess?

First, Second and Next-Generation Systems

A review of the implemented system literature reveals the existence of several competing knowledge-based system technologies, which can generally be differentiated by their architecture and capabilities [11]. For illustrative purposes (not to be construed as a definitive classification), these technologies can be categorized as first, second and next-generation systems (Table 2).

Table 2. Knowledge-Based System Technologies

	First Generation	Second Generation	Next-Generation
Architecture	Rule Based [18]	Reasoning or Control Strategies are Explicitly Represented [18]	Integrative with Multilevel Modular Components [10]
System Capabilities - Scale	- Stand-alone Systems for Solving Small Static Problems [10, 18]	- Stand-alone or Integrated Systems for Small/Medium size Problems	- Cooperative Real-time Solution of Dynamic Large-Scale Problems
- Management of Uncertainty	- Rigid (add on) techniques for the management of uncertainty [11]	- Sophisticated Uncertainty Management Approaches [18]	- Multiple Uncertainty Management Approaches [10]

Implications: A Military Perspective

The application of knowledge-based system technology to military problems is an ongoing and important endeavour. Anyone who doubts this estimate of the situation need only review the events of the Gulf War, which was an information war [25], and take the time to piece together how much of the war was directly influenced by knowledge-based systems [25, 26]. These systems had a major operational impact on transportation planning [1], were instrumental in planning air strikes [29], provided commanders with unprecedented command and control networks [25], and enhanced combat effectiveness through the provision of near real-time information [26]. In short, knowledge-based systems have been applied to most of the component parts of the military, and this technology has the potential to affect the entire military problem domain (Figure 5).

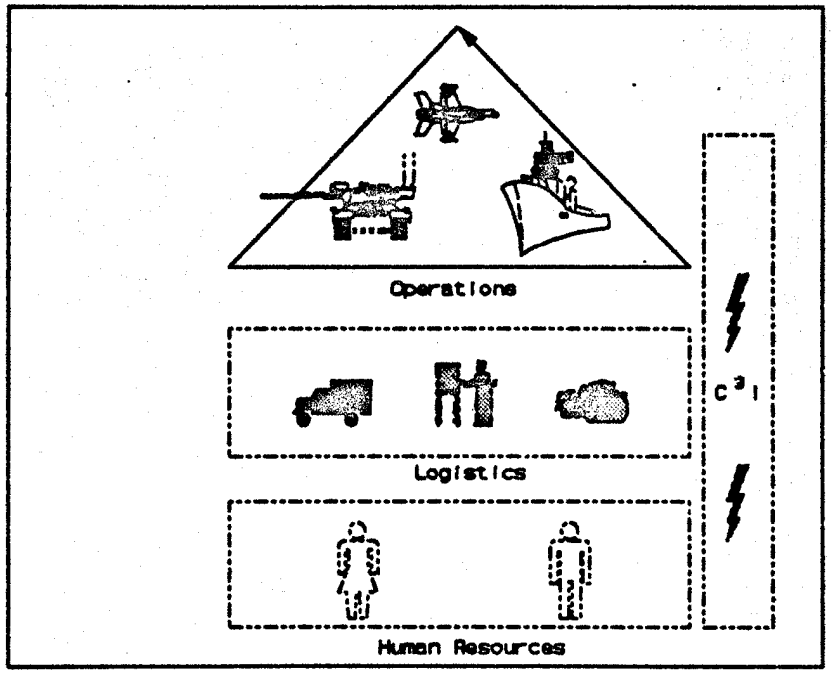


Figure 5. The Military Problem Domain

Opportunity or Threat

In an increasingly complex world, where accuracy and the requirement to process an exploding mass of information make computers a fact of life, automated systems pose both an opportunity and a threat [25]. This is not to say that knowledge-based systems offer a panacea for all military problems, or that failures have not occurred [30]. However, there is ample evidence to suggest that this technology, where wisely applied, can lead to a more efficient and effective military, as discussed in the following case study of the Bayesian Reasoning Tool.

The Bayesian Reasoning Tool: A Case Study

The Bayesian reasoning tool (BaRT), is a knowledge-based system that is currently under development at the Naval Research Laboratory (NRL), Washington, as an aid in classification problem solving [24, 33, 34]. A case study of BaRT was conducted, and forms part of this paper for three reasons: first, BaRT has been successfully applied to two military problems (classifying ship images, and analyzing intelligence reports [24]); secondly, BaRT's architecture permits stand-alone or cooperative operation (Figure 6); and last, but not least, a wealth of information on BaRT was available through the XTP-4 research community.

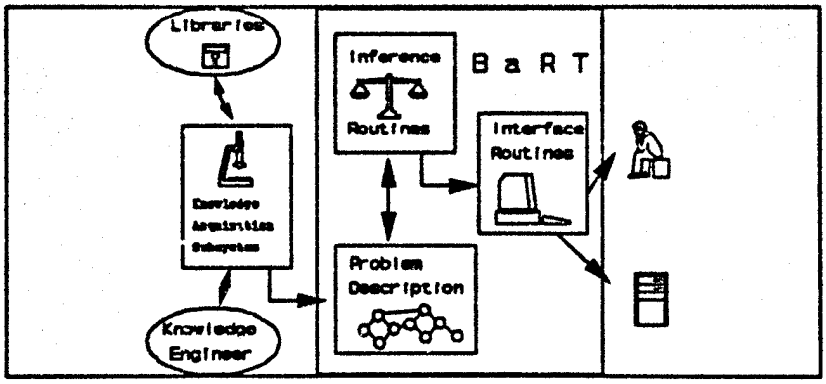


Figure 6. BaRT System Architecture [Modification of 24, p. 277]

Background

BaRT was originally developed as a prototype ship classification aid, which is a complex problem that can have life-and-death consequences (see [24] for a discussion of this problem). Initial work was based on the PROSPECTOR updating method, and a representative subset of 10 naval classes was selected from the 600 + naval classes that comprise the operational environment. Extensive testing, by Navy experts, clearly demonstrated this system's capabilities, which were judged to be excellent (using realistic test images [24]). As a result of this success, BaRT was subsequently modified to meet operational Navy requirements, and the most recent version of BaRT supports three different uncertainty management approaches: Bayesian networks, influence diagrams, and taxonomic hierarchies [33]. This version of BaRT was also tested, using the same test data as the original prototype, and almost identical test results were obtained using a less complicated model (a 36 node network with 35 links, compared to the PROSPECTOR version which required 181 nodes and 297 links [24]).

Bayesian Belief Networks

Belief networks are one of the formalisms that have been proposed for the management of uncertainty that are considered appropriate for hierarchical classification problems, and these networks form an important component of the knowledge representation and reasoning schemes in BaRT. Although a number of different Bayesian network approaches exist (variously referred to as influence diagrams, causal nets, belief nets, knowledge maps, relevance diagrams, and Bayesian networks [11, 35]), the approach implemented in BaRT is based on the contention that hypotheses and relations in a given domain can be represented in a network of low order probabilistic relationships between clusters of semantically related hypotheses [36]. These networks consist of a set of exhaustive and mutually exclusive hypotheses, $H=\{h_1, h_2,...h_n\}$ which are quantified with numerical values that indicate the degree of belief accorded to each hypothesis by some

body of knowledge K. This is illustrated in Figure 7 using the IFF problem as an example (see [11] for a detailed discussion), where h_1 represents the statement "the approaching unidentified agent is hostile," and $\mathcal{P}(h_1|K)$ denotes the subjective belief in h_1 given a body of knowledge K (which is influenced by the set of hypotheses $H = \{h_1, h_2, h_3, h_4, h_5\}$). The links between the nodes, which are outlined in Table 3, indicate causal, probability, relationships. To fully specify this network eleven conditional probability values are required, instead of the standard 2^n-1=31, due to the Bayesian network independence assumption [11].

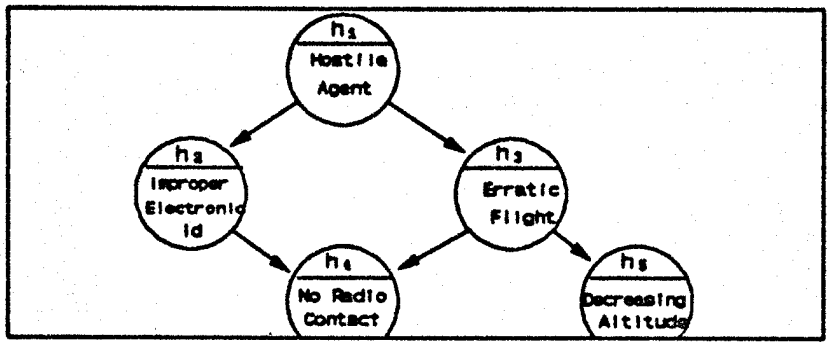


Figure 7. Directed Graph for the IFF Domain Model

Table 3. Conditional Probabilities for Figure 7

Attribute	Value	Explanation
$\mathcal{P}(h_1)$	0.20	Current intelligence estimate
$\mathcal{P}(h_2\|h_1)$ $\mathcal{P}(h_2\|\neg h_1)$	0.80 0.20	Improper electronic id is uncommon, but frequent if agent is hostile
$\mathcal{P}(h_3\|h_1)$ $\mathcal{P}(h_3\|\neg h_1)$	0.20 0.05	Erratic flight is rare, and uncommon consequence of hostile agent
$\mathcal{P}(h_4\|h_2 \wedge h_3)$ $\mathcal{P}(h_4\|h_2 \wedge \neg h_3)$ $\mathcal{P}(h_4\|\neg h_2 \wedge h_3)$ $\mathcal{P}(h_4\|\neg h_2 \wedge \neg h_3)$	0.80 0.80 0.80 0.05	Absence of radio contact is rare, but frequent if either improper electronic id or erratic flight present
$\mathcal{P}(h_5\|h_3)$ $\mathcal{P}(h_5\|\neg h_3)$	0.80 0.60	Decreasing altitude is common, but frequent when flight is erratic

Once the causal relationships between the hypotheses that form the domain model illustrated in Figure 7 have been identified (Table 3), this information can be represented in an undirected graph (Figure 8), composed of a number of cliques $\{h_1, h_2, h_3\}$, $\{h_2, h_3, h_4\}$, and $\{h_3, h_5\}$ which are assumed to be independent of all those hypotheses that are not adjacent, conditional on those hypotheses that are adjacent [36, 38]. These hypotheses are specified by the marginal distributions derived from the conditional probabilities shown in Table 3, according to (1), as illustrated by (2) and (3), and are stored in Table 4 for future reference.

$$\mathcal{P}(h_1 \wedge h_2 \wedge h_3 \wedge h_4 \wedge h_5) = \frac{\mathcal{P}(h_3 \wedge h_5)}{\mathcal{P}(h_3)} \cdot \frac{\mathcal{P}(h_2 \wedge h_3 \wedge h_4)}{\mathcal{P}(h_2 \wedge h_3)} \cdot \mathcal{P}(h_1 \wedge h_2 \wedge h_3) \quad (1)$$

$$\mathcal{P}(h_1 \wedge h_2 \wedge h_3) = \mathcal{P}(h_2|h_1)\,\mathcal{P}(h_3|h_1)\,\mathcal{P}(h_1) = 0.8 \times 0.2 \times 0.2 = 0.032 \quad (2)$$

$$\mathcal{P}(\neg h_1 \wedge h_2 \wedge h_3) = \mathcal{P}(h_2|\neg h_1)\,\mathcal{P}(h_3|\neg h_1)\,\mathcal{P}(\neg h_1) = 0.2 \times 0.05 \times 0.8 = 0.008 \quad (3)$$

etc. ...

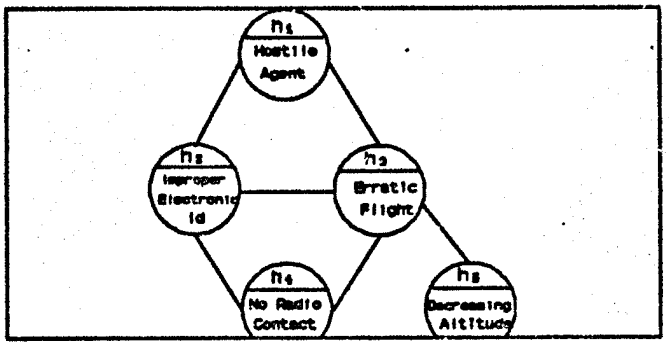


Figure 8. Undirected Graph for the IFF Domain Model

Table 4. Marginal Distributions on the Cliques of Figure 8

Clique $\{h_1, h_2, h_3\}$	Clique $\{h_2, h_3, h_4\}$	Clique $\{h_3, h_5\}$
$\mathcal{P}(h_1 \wedge h_2 \wedge h_3) = 0.032$	$\mathcal{P}(h_2 \wedge h_3 \wedge h_4) = 0.032$	$\mathcal{P}(h_3 \wedge h_5) = 0.064$
$\mathcal{P}(\neg h_1 \wedge h_2 \wedge h_3) = 0.008$	$\mathcal{P}(\neg h_2 \wedge h_3 \wedge h_4) = 0.032$	$\mathcal{P}(\neg h_3 \wedge h_5) = 0.552$
etc. ...	...	...
Total = 1.000	Total = 1.000	Total = 1.000

From Table 4, prior probabilities for each hypothesis can be derived (i.e., $\mathcal{P}(h_5) = \mathcal{P}(h_3 \wedge h_5) + \mathcal{P}(\neg h_3 \wedge h_5) = 0.064 + 0.552 = 0.616$, $\mathcal{P}(h_4)=0.320$, $\mathcal{P}(h_3)=0.080$, $\mathcal{P}(h_2)=0.320$, and $\mathcal{P}(h_1)=0.200$), and from this information it is possible to assess the impact that new evidence will have on the probability of any other hypothesis. For example, propagating the observation that hypothesis h_5, decreasing altitude is true, will lead to a number of revisions of belief. This involves the re-ordering the cliques $\{h_3, h_5\}$, $\{h_1, h_2, h_3\}$, and $\{h_2, h_3, h_4\}$ as shown in Figure 9, and propagating the new evidence through the re-ordered network in accordance with (4) and (5). Which, eventually leads to the calculation of revised marginal probabilities, as follows: $\mathcal{P}^*(h_1)=0.208$, $\mathcal{P}^*(h_4)=0.333$, and $\mathcal{P}^*(h_2)=0.325$.

Current belief $\mathcal{P}^*(h_3)$ = prior belief given some evidence $\mathcal{P}(h_3|h_5)$ (4)

or $\mathcal{P}^*(h_3) = \mathcal{P}(h_3|h_5) = \frac{\mathcal{P}(h_3 \wedge h_5)}{\mathcal{P}(h_5)} = \frac{0.064}{0.616} = 0.104$ from Table 4 (5)

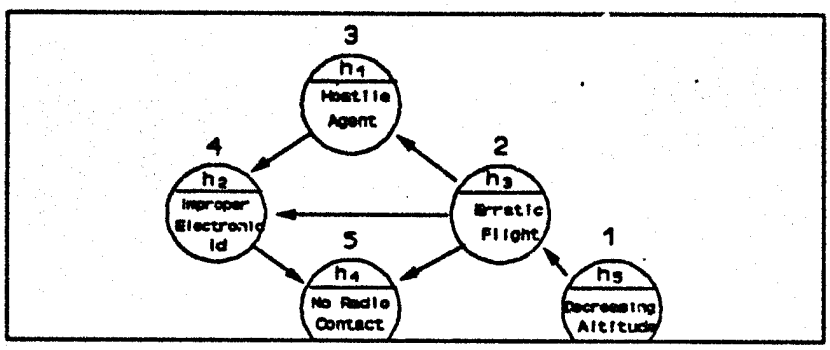


Figure 9. Assessment of the effect of a Decrease in Altitude

Although the above example may appear to be overly simplified, there has been at least one highly publicized instance of an unidentified agent being mistakenly classified as hostile (by the USS Vincennes) based on a fairly similar scenario [4]. In any event, the completeness of domain model should not be the issue, what is important to note is that network representations can effectively be employed to manage the uncertainty inherent in solving classification problems.

Applying BaRT to Military Problems

BaRT has been successfully applied to a number of military problems [32], and work aimed at applying BaRT to other military problems is currently under way. These efforts have been aided by the fact that BaRT incorporates a number of different knowledge representation and reasoning schemes, and the ability of BaRT to operate as a stand-alone classification decision aid, or as a cooperative component of a larger knowledge-based system [32]. The importance of these characteristics cannot be overstated (see [11] for further details), and particular note should be made of the fact that reusable knowledge-based system components can be developed and applied to military problems [11].

Although details are sketchy, the module-oriented approach to problem solving employed by BaRT appears to adhere to the development framework that forms the basis for much of the recent DARPA sponsored research aimed at developing next-generation knowledge-based systems [1, 10, 22, 26, 31], like the Pilot's Associate. A comprehensive analysis of this research, which encompasses the entire spectrum of military problems, would undoubtedly assist any military professional seeking to exploit the potential of knowledge-based systems technology. However, few technical details are available [11], and much of this research is shrouded in secrecy. For this reason an analysis of BaRT is particularly useful to the military professional seeking a better understanding of the problems that can affect the implementation of this technology [30].

BaRT is an automated reasoning tool, developed at the Naval Research Laboratory (NRL), which employs a framework of Bayesian techniques to manage the uncertainty associated with generic classification problems. BaRT can function as a stand alone decision aid, or it can act as the classification module of a larger knowledge-based system, and BaRT has been successfully applied to a number of military classification problems.

Conclusion

Knowledge-based systems have evolved to the point where they can coherently represent large amounts of knowledge and efficiently reason with uncertain information [10]. These same systems have proven their worth in industry [17-20], and during the Gulf War [25, 26], and the implications of this technology on the military are clear: evolution, or extinction.

Although no clear "best" approach can be identified for the management of uncertainty in knowledge-based systems, various techniques have proven their applicability to specific types of problems [8, 11]. Efforts aimed at developing task-specific systems have led to the realization that many problems can be decomposed into task-specific component parts (e.g., the classification problem addressed by BaRT), and that knowledge-based systems capable of solving these generic tasks can be used to construct cooperative systems. A number of these cooperative knowledge-based systems are being developed to address complex military problems [22, 31], and development frameworks employing a module-oriented approach have been developed to facilitate this process [27].

Next-generation knowledge-based systems, with multilevel application architectures, have demonstrated that feasible solutions to complex problems can be obtained [31], and further developments are expected in the near future [10]. The implementation of these systems can result in failure [30], and it is imperative that organizations wishing to adopt this technology adopt a systematic approach that address both technical and organizational concerns.

Notes

1. The terms "knowledge-based system" and "expert system" are not used by all authors in the same context, and the expression knowledge-based system is used throughout this text, in accordance with [16], to refer to any computer system that consists of a knowledge base, and a knowledge base manager.

2. While acknowledging that efforts aimed at developing computer systems capable of providing military commanders with intelligent assistance can be viewed from a number of different perspectives, and that many of these views are based on radically different philosophic and moral beliefs, this paper does not address these issues. This text is not a philosophic dissertation, it examines what has been reported in the open literature, and no attempt has been made to resolve any of the underlying philosophical problems with, or the moral objections to, the development of intelligent advisors for military use.

3. Although a comprehensive review of the uncertainty management literature would clearly be useful, there are a number of practical problems which make such an undertaking a virtual impossibility [16]. These problems include: the quantity of information [5, 6], the diversity of uncertainty management approaches [11], the range of practical techniques that exist [15], and the complexity of the knowledge-based system environment. In addition, uncertainty management approaches are designed to deal with the multidimensional problem of acquiring, representing, and reasoning with knowledge, and each of these issues are complex problems in their own right.

References

[1] Amarel, S., AI research in DARPA's strategic computing initiative, IEEE Expert, June, 1991, pp. 7-11.

[2] Hink, R.F., & Woods, D.L., How humans process uncertain knowledge: An introduction for knowledge engineers, AI Magazine, 1987, pp. 41-53.

[3] Kleinmuntz, B., Why we still use our heads instead of formulas: Toward an integrative approach, Psychological Bulletin, Vol. 107, 1990, pp. 296-310.

[4] Neumann, P.G., Inside risks: The human element, Communications of the ACM, Vol. 34, No. 11, 1991, p. 150.

[5] Carande, R., Checking out AI sources, AI Expert, Vol. 3, No. 6, June, 1988, pp. 60-65.

[6] Lirov, V., & Lirov, Y. A subject bibliography of logic programming applications in control and decision support systems, Computers Math, Applications, Vol. 20, No. 9/10, 1990, pp. 141-179.

[7] Ng, K.C., & Abramson, B., Uncertainty management in expert systems, IEEE Expert, April, 1990, pp. 29-48.

[8] Chandrasekaran, B., & Tanner, M.C., Uncertainty handling in expert systems: Uniform vs. task-specific formalisms. In L.N. Kanal, & J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence, Amsterdam, Holland, 1986, pp. 35-46.

[9] Grzymala-Busse, J.W., Managing Uncertainty in Expert Systems. Kluwer Academic Publishers, Norwell, MA, USA, 1991.

[10] Mark W.S., and Simpson R.L., Knowledge-based systems an overview, IEEE Expert, June 1991, pp. 12-17.

[11] McKean, M.C., Management of uncertainty in knowledge-based systems: A survey, Unpublished Master's Thesis at the Royal Military College of Canada, Kingston, ONT, CAN, 1992.

[12] Shafer, G., & Pearl, J. (Eds.), Readings in Uncertain Reasoning. Morgan Kaufmann Publishers, Inc., San Mateo, CA, USA, 1990.

[13] Allen, J.F., Kautz, H.A., Pelavin, R.N., & Tenenberg, J.D., Reasoning About Plans. Morgan Kaufmann, San Mateo, CA, USA, 1991.

[14] Anderson, J.R., Cognitive Psychology 'and Its Implications' (3rd Ed.). W.H. Freeman and Company, New York, NY, USA, 1990.

[15] Chandrasekaran, B., Interview: Frederick Hayes-Roth and Richard Fikes, IEEE Expert, October, 1991, pp. 3-14.

[16] Davis, E., Representations of Commonsense Knowledge. Morgan Kaufmann Publishers, Inc., San Mateo, CA, USA, 1990.

[17] Dym, C.L., & Levitt, R.E., Knowledge-Based Systems in Engineering. McGraw-Hill, Inc., New York, NY, USA, 1991.

[18] Johnson, N.E., Tomlinson, C.M., & Johnson, L., Second generation expert systems and knowledge elicitation, International Journal of Systems Research and Information Science, Vol. 4, 1990, pp. 87-99.

[19] Mavis, R., & Keyes, J. (Eds.), Handbook of Expert Systems in Manufacturing. McGraw-Hill, Inc., New York, NY, USA, 1991.

[20] McGhee, J., Grimble, M.J., & Mowforth, P. (Eds.), Knowledge-Based Systems for Industrial Control. Peter Peregrinus Ltd., 1990.

[21] Rich, C., Letter from the guest editor, SIGART Bulletin (Special Issue), Vol. 2, No. 3, June, 1991.

[22] Andricle, S.J. (Ed.), Artificial Intelligence and National Defense: Applications to C^3I and Beyond. AFCEA Press, Washington, D.C., USA, 1987.

[23] Banks S.B., & Lizza C.S., Pilot's associate a cooperative, knowledge-based system application, IEEE EXPERT, June, 1991, pp 18-29.

[24] Booker, L.B., & Hota, N., Probabilistic reasoning about ship images. In J.F. Lemmer, & L.N. Kanal (Eds.), Uncertainty in Artificial Intelligence, Vol. 2, Amsterdam: North-Holland, 1988, pp. 371-379

[25] Campen, A.D. (Col Ret.), Gulf war's silent warriors bind U.S. units via space, SIGNAL, August, 1991, pp. 81-84.

[26] Cheadle, G. (BGen Ret.), Commanders garner wallop from new system advances, SIGNAL, November, 1991, p. 71.

[27] Hayes-Roth, F., Davidson, J.E., Erman, L.D., & Lark, J.S., Frameworks for developing intelligent systems: The ABE systems engineering environment, IEEE Expert, June, 1991, pp. 30-40.

[28] Robinson, C.A., Information explosion adds punch to tactical weapons, SIGNAL, September, 1991, pp. 25-29.

[29] Signal, Aerial combat stimulates Saudi air force blueprint, SIGNAL, August, 1991, pp. 116-117.

[30] Sloane, S.B. (Capt(N) Ret.), The use of Artificial Intelligence by the United States Navy: Case study of a failure, AI Magazine, 1991, pp. 80-92.

[31] Tanner, M.C., & Keuneke, A.M., Explanations in knowledge systems: The roles of the task structure and domain functional models, IEEE Expert, June, 1991, pp. 50-57.

[32] Booker, L.B., & Ramsey, C.L., Classification problem solving using Bayesian techniques. Proceedings of the Sixth Annual Workshop on Command and Control Decision Aiding, San Diego, CA, USA, February, 1989.

[33] Booker, L.B., Hota, N., & Ramsey, C.L., BaRT: A Bayesian reasoning tool for knowledge based systems. In M. Henrion, R.D. Shachter, L.N. Kanal, & J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence, Vol. 5, Amsterdam: North-Holland, 1990, pp. 271-282.

[34] Hota, N., Ramsey, C.L., Chang, L.W., & Booker, L.B., BaRT Manual Version 3.0, Naval Research Laboratory, Washington, DC, USA, February 14, 1991.

[35] Neapolitan, R.E., Probabilistic Reasoning in Expert Systems: Theory and Algorithms. John Wiley & Sons, Inc., New York, NY, USA, 1990.

[36] Pearl, J., Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers, Inc., San Mateo, CA, USA, 1988.

[37] Shachter, R.D., Evaluating influence diagrams, Operations Research, Vol. 34, No. 6, November-December 1986, pp. 871-882.

[38] Spiegelhalter, D.J., Probabilistic reasoning in predictive expert systems. In L.N. Kanal, & J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence. Elsevier Science Publishers B.V., New York, NY, USA, 1986, pp. 47-67.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

KNOWLEDGE ENGINEERING FOR PACES, THE PARTICLE ACCELERATOR CONTROL EXPERT SYSTEM

P.C. Lind[1], W.F.S. Poehlman[2], J.W. Stark[3], T. Cousins[4]

Abstract

The Particle Accelerator Control Expert System (PACES) is an expert-system-based real-time control system for the KN-3000 high voltage particle accelerator. It is designed to be an accelerator operators companion, automatically performing established procedures, such as start-up and shut-down, in order to relieve the operator from the drudgery of accelerator control. Additionally, it will act as an accelerator performance monitor with the ability to rapidly detect operating faults or anomalies, alert the operator, and suggest possible solutions. PACES is planned also to serve as an educational platform for the training of new accelerator operators.

This paper is concerned in particular with the knowledge engineering aspects of the system. After attempting to implement the knowledge base under two different commercial shells, it was realized that neither offered an ideal development environment. The chosen course of action was to develop a customized shell that would provide both flexibility and inferencing speed. Both of these factors are crucial to effective real-time control of the accelerator.

Knowledge engineering for PACES involves several important issues. First, there is the need to encapsulate expert knowledge into the system in a form that facilitates automatic accelerator operation. This is a generic problem in expert system design whose solution usually differs from application to application. Second, it is necessary under certain circumstances to partition the system in such a way that time-consuming inferencing is minimized in favour of faster, more algorithmic control. It is seen that accelerator control will require fast, narrow-minded decision-making for rapid fine tuning, but slower, broader reasoning for machine start-up, shut-down, and fault diagnosis and correction. Additionally, it is important to render the knowledge base in a form that is conducive to operator training; the expert system must be aware of different levels of accelerator expertise and respond accordingly.

A promising form of PACES involves a hybrid system in which high-level reasoning is performed on the host machine that interacts with the user, while an embedded controller employs neural networks for carrying out fast but limited on-the-fly adjustment of accelerator performance. This partitioning of duty facilitates a hierarchical chain of command, yielding an effective mixture of speed and reasoning ability.

[1]Dept of Electrical and Computer Engineering, [2] Dept of Comp Sci and Systems, [3]McMaster Accelerator Lab, McMaster Univ, Hamilton, [4]Scientific Authority, Electronics Div, DREO, Ottawa

Introduction

The KN model 3000 particle accelerator [5] situated at the Defence Research Establishment Ottawa (DREO) is employed primarily to produce monoenergetic neutrons for the calibration of radiation detectors. Its daily use involves little variation since, at present, it is only operated for this single purpose. Recently, many accelerator operators (experts) have reached retirement age and there is a need to transfer their operational expertise to their successors. In an effort to smooth this transition, and to provide a solid foundation for new operators, it was deemed productive to develop an expert system for operation of the KN. This expert system would not only provide assistance for new operators, but would also reduce the operator time required for accelerator start up, steady-state control and shut-down, allowing other more valuable tasks to be performed.

The requirements of the overall control system are summarized below (in no particular order):

- ➢*Robustness*: The accelerator is a source of both high-voltage and moderate radiation levels, and the system must be resistant to such hazards.
- ➢*Manual override*: At any time, the human operator must be able to rapidly assume complete, direct control of the accelerator.
- ➢*User-friendliness*: The front end (user interface) of the system must be both easy and desirable to use (so that the user finds the system helpful and beneficial).
- ➢*"Smarts"*: Although the system is presided over by a human operator, it must still be able to act autonomously, making decisions in a fashion that mimics the expert operator as closely as possible.
- ➢*Speedy decision making*: Control of the accelerator is a *real-time* control problem, and, as such, the control system is time-critical.
- ➢*Fault detection, analysis and diagnosis*: A major aspect of the control system's knowledge base is the instillation of the expert's knowledge of how to detect operating faults, analyze them and suggest and/or affect remedial action.
- ➢*Repeatability*: The control system should be able to reproduce previous operating parameters in order to repeat past particle beam behaviour.
- ➢*Flexibility and Extendibility*: The control system must facilitate easy upward expansion.

To these ends, the Applied Computersystems Group [6] (ACsG) has undertaken to develop an expert-system-based accelerator control facility, primarily for use with the

KN-3000 at DREO [7]. This control system, the Particle Accelerator Control Expert System (PACES), is intended to augment operation by raising the operational expertise of non-expert operators. Thus, an operator will be relieved of the repetitive aspects of accelerator control.

For a more detailed presentation of the initial project description and a summary of early development, please refer to [8].

Overview of the Operating Environment

The KN is a Van de Graaff particle accelerator that uses the generation of static charge to provide particle energies up to 4 MeV (mega-electron volts). Panel 1 depicts a schematic of the accelerator [9], showing the Van de Graaff generator housed inside a tank that is electrically insulated with pressurized sulphur-hexafluoride (SF_6). A radio frequency ion source (typically hydrogen or deuterium) is used to inject particles into the accelerator. The particles are accelerated from the machine in a beam which can be focused and steered to one or more target areas where the actual experiments occur.

Usual operation of the accelerator begins with the machine in a "cold" state (i.e. completely inactive). The operator carries out a start-up procedure (which may call for the re-establishment of a previous day's operating conditions) that culminates in the generation of a stable particle beam that is operated in a steady-state mode (or perhaps pulsed mode) for a length of time (typically several hours), after which the machine is shut down to its "cold" state once again. Aside from daily variations in operating conditions, start-up is essentially the same process each time, whereas the shut-down procedure seldom varies. Steady-state operation, however, depends largely on the nature of the day's experiments as well as the dynamic behaviour of the machine itself (eg. changes in behaviour due to heating, age and condition of components). It may be necessary during this steady-state period to perform small parameter changes, such as altering particle energy, beam current or ion species. The steady-state operation is further characterized by the possible occurrence of operating anomalies, faults and/or failures.

Panels 2 and 3 show a stylized image of the accelerator's control panel. This control panel presents three main types of instrumentation: meters, selsyns and switches. The meters register the various operating parameters of the accelerator (e.g. vacuum pressure, beam current, terminal voltage). The selsyns are pairs of electrically coupled dynamos, one

on the control panel and one inside the accelerator tank, that enable the operator to manually adjust a physically isolated mechanism (e.g. a rheostat or valve) via an electrical feedback link. The switches turn on or off, for example, the control power and the drive motor of the Van de Graaff generator, and select the type of gas for the ion source.

During normal operation, the operator watches the meters to observe the behaviour of the accelerator and adjusts some or all of the four selsyns to alter this behaviour to a more suitable state. Thus, from a process control perspective, the meters are *inputs* to the controller, while the selsyns (and switches) are *outputs* of the controller.

Control System

PACES is a complex, distributed control program that runs on two or more computers (Figure 1). The *host* computer, typically an AT-style 386 or 486 computer with high resolution SVGA or XVGA graphics capability, runs the front end user interface and expert system. The host computer is connected through an RS-232 serial line to a single board computer [10] (SBC) embedded controller. This SBC is connected to the accelerator through special electronics circuitry, and is responsible for direct accelerator control and data acquisition. The host computer acquires accelerator operating parameters from the SBC and uses them to make control decisions which result in control commands being sent to the SBC. The combination of host computer, SBC and control circuitry constitute the control system's closed feedback loop.

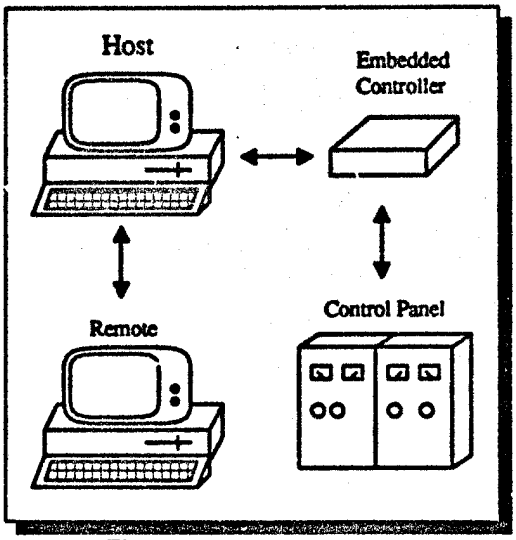


Figure 1. PACES components.

The host computer may also be connected to one or more *remote consoles*, typically AT-style 386 or 486 computers. The remote consoles are connected to the host via RS-232 serial links, and may use high speed (9600 baud) modems to establish links over telephone lines. The remote consoles act as functional extensions of the host computer's user interface, enabling the accelerator operator to

monitor and control the machine from locations other than the main control room. This is useful during experiments, when the experimenters spend most of their time in the target area, but still wish to have control over the accelerator without having to travel to the control room. Additionally, the use of a modem over telephone lines enables a person at another location, perhaps many miles away, to access and operate the accelerator for expert diagnosis or troubleshooting purposes.

The control system can be roughly divided into five sections (Figure 2). The *system manager* is the top level of the program that oversees and coordinates the other sections. The *user interface* is the control system's link with the operator. The *control interface* is responsible for uploading telemetric data from the SBC and sending it control commands. Finally, the *inference engine* and *knowledge base* are used for making heuristic decisions.

PACES runs under Microsoft Windows 3.0 [11]. It is written in Turbo Pascal for Windows [12] (TPW), which offers an easy-to-use object-oriented platform for rapid, straightforward development of Windows applications.

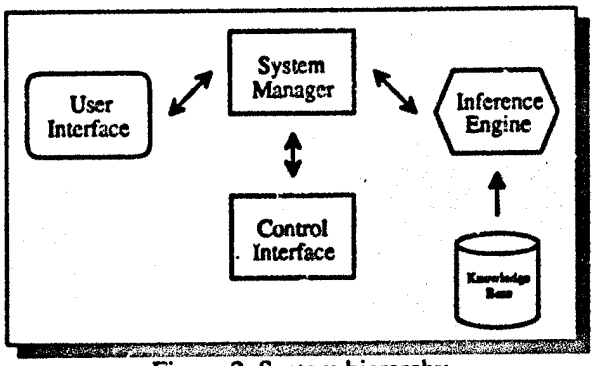


Figure 2. System hierarchy.

User Interface

The Windows graphical user interface (GUI) is employed to present the accelerator operator with a functional facsimile of the accelerator's control panel (Panels 2 and 3). Effort has been taken to preserve as much of the real control panel's appearance and behaviour as possible. This control panel enables the operator to control the machine through the computer by reading analog meters and using the computer's mouse and keyboard to perform control actions. The operator can also initiate automated start-up or shut-down procedures, browse and repeat past accelerator runs, and log accelerator data to disk.

Graphical *meters* display real time accelerator operating parameters such as terminal voltage and beam current. It was found during preliminary field tests that accelerator

operators, being used to reading real analog meters, preferred them to strictly digital displays. The PACES meters provide both analog and digital displays, so that the operator sees the usual analog display as well as a quantitative value. Additionally, each of the meters has a "stripchart" graph (see Panel 3) that can be opened to display the meter's value and rate of change as functions of time.

The primary controls for the accelerator are the four *selsyns* (shown as dark circles on the control panel in Figure 2). These selsyns may be operated manually or set on automatic; in automatic mode, the control system uses algorithmic and heuristic techniques to alter the selsyn settings in order to maintain particle beam stability.

There are also various *buttons* and *switches*, some that directly control the accelerator (e.g. power on/off, drive motor on/off), and others invoking sub-systems of the program (such as the start-up procedure).

Expert System

Requirements

Four key areas of accelerator control rely on classical knowledge-based inferencing:

- Start-up: The expert system is used to start the accelerator and produce a stable particle beam with specified physical characteristics.
- Shut-down: The accelerator is shut-down in an expedient manner. As described in the section on Fault Detection and Diagnosis, the expert system may be called upon to perform normal shut-down, rapid safing (in order to recover from a possibly dangerous anomaly), or panic shut-down (in event of a catastrophic fault).
- Steady-state: The particle beam must be maintained within the specified operating limits. The expert system is used to maximize beam stability while affording the operator some capability to alter the accelerator's behaviour as required.
- Fault Detection and Diagnosis: The knowledge base and expert system are used to detect and diagnose faults in the accelerator. Detection involves expert heuristics to determine from the accelerator operating parameters what type of fault or aberration is occurring. Diagnosis involves selection of the best means of overcoming the problem, be it a simple control adjustment, immediate shut-down or request for operator intervention.

The "expert system" component of PACES is, consequently, required to posses several important characteristics:

- ➢ It must be *embedded* in the control program as a whole, and not a stand-alone top level shell. Due to the real time nature of the control system, it is necessary to relegate the "expert system" to a subservient level; the expert system is invoked by the system manager when needed, but never assumes complete control. The expert system is periodically called upon to detect operating faults or anomalies and to perform heuristic decision making.
- ➢ It must provide both forward and backward chaining. Forward chaining is used during steady-state beam maintenance, while backward chaining is used during fault detection/recovery and when the operator desires to alter beam characteristics.
- ➢ The expert system must be amenable to "stop and start" inferencing. The real time nature of the control problem requires that the inference engine be able to suspend its current reasoning path in order to follow some other reasoning path as events warrant.

Early Attempts

Two early attempts at integrating an "off the shelf" expert system shell with PACES met with disappointing results. The Personal Consultant Plus (PC+) [13] expert system shell was considered first, but was found to be quite unamenable to being embedded in a larger application [8]. Its snail-like inferencing speed, high demand on system resources and complicated external language interface mechanism made it unwieldy as an "on line" real time embedded inference engine. Kappa-PC [14] was also briefly evaluated since it offers many features that PC+ lacks. Kappa-PC is designed as a Microsoft Windows-based extensible expert system shell. Although Kappa-PC was, in many ways, ideal for the requirements at hand [8], it is, at present, only extensible using Microsoft C, which was found to possess a somewhat hostile user interface aversive to rapid software prototyping. A desirable form of Kappa-PC would be that of a Windows-format dynamic link library (DLL) that could easily be integrated into the control system regardless of the programming language used for development.

It was finally decided that the best approach was to use a custom-made expert system shell that would be designed from the outset to possess the desired qualities. The next section briefly describes this shell, the *Windows Application eXpert system* (WAX) shell.

Expert System Design

The WAX shell is written as a collection of objects in a TPW unit (separately compiled module). Figure 3 shows how WAX is integrated into the application as a whole. The WAX unit does not contain any "knowledge", but only the mechanisms for manipulating (inferencing) knowledge. The "knowledge bases" are stored in separate TPW units, and dynamically linked to the inference engine at runtime. These knowledge base units, which achieve the useful software engineering property of *information hiding*, contain *rules* and *information objects* (object-oriented data structures). In developing knowledge objects, the knowledge engineer (programmer) is free to use as much of TPW's extensible object-oriented capability as desired. The knowledge base rules are written by the knowledge engineer to process the information objects in order to derive new information objects or cause side effects in the application. (Two examples of side effects are a pop-up message prompt that asks the user to input a piece of information, and assertion of a control action that causes actuation of an external device.)

The principal object in the WAX unit is the inference engine that is used to evaluate rules.. Rules are passed to the inference engine in prioritized *rule sets* that represent domains of knowledge. The inference engine is re-entrant so that it can be interrupted while inferencing one rule set in order to inference a different rule set. Rules are able to change the course of inferencing by calling WAX subroutines that start, suspend or stop inferencing chains, or alter the membership of rule sets.

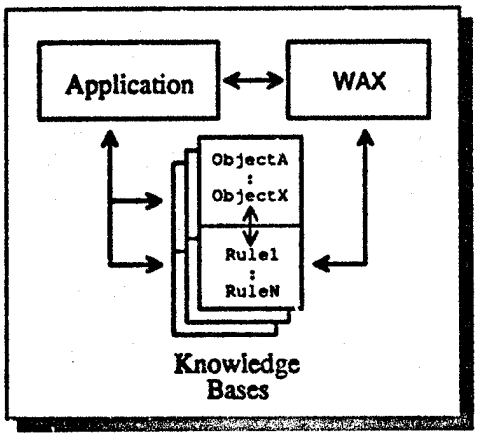


Figure 3. The WAX shell.

Typically, the top level of the application initiates inferencing in response to a request from the user, or after some external condition has occurred (such as a timed event or interrupt). The application dynamically creates an inference engine object and passes it a rule set, thereby starting the inferencing mechanism. The inference engine then proceeds to evaluate the rules in its rule set until the rule set is empty or the engine is suspended or

stopped. Inferencing proceeds under the Windows multitasking environment so that the inferencing can be performed "in parallel" with other operations [15].

The WAX implementation results in a flexible inferencing system that places a large burden on the knowledge engineer with regard to implementing the knowledge base. In certain situations, this approach is unsuitable since it requires that the knowledge engineer be a proficient programmer. Nevertheless, WAX is intended primarily as an embeddable inferencing system, and thus, is inherently tied with software development and programming. WAX is not meant to replace conventional shells that offer ease of knowledge encapsulation without requiring a high level of programming ability on the part of the knowledge engineer. Instead, WAX is designed to sacrifice generic ease of use and high shell overhead in favour of an extreme amount of low level flexibility and minimal shell overhead.

Knowledge Engineering for PACES

The problem of controlling the accelerator presents a myriad complications, ranging from real time control considerations and controller robustness to educational and safety issues. The PACES knowledge base should encapsulate as much expert knowledge as possible, especially in the area of fault detection and diagnosis, while still preserving a high degree of real time response.

Knowledge engineering for PACES was performed in conventional manner. Various expert accelerator operators were interviewed and observed over several sessions, notes were taken by hand and the interviews were taped using an audio cassette recorder. Following preliminary interviews, a prototype knowledge base was constructed that implemented the start-up and shut-down procedures. A development cycle ensued in which testing was followed by expert evaluation and knowledge base modification and further testing. At present, the knowledge base is still in this development phase.

Algorithmic Control vs. Heuristic Decision Making

The control system as a whole is partitioned into several layers. The SBC is the lowest level, possessing little or no heuristic knowledge, but a large amount of algorithmic procedures [16]; the SBC responds to control and data acquisition commands, but has no

volition of its own. At the top level, the host PC strikes a balance between complete heuristic activity and algorithmic activity. Algorithms are used for well defined control procedures in order to increase efficiency and response time, whereas heuristics (inferencing) are used for expert-like decision making. Certain aspects of the accelerator control problem are more suited to being implemented as algorithms. Shut-down, for example, almost always follows the same, step-by-step procedure. Steady state beam maintenance, however, calls for a mix of heuristics and algorithms; control decisions are made using inferencing, but execution of the control decisions (e.g. feedback loops) requires algorithmic processing. The WAX shell provides a simple means of melding procedural activity with heuristic reasoning: Since WAX rules are merely external procedures, they are unconstrained as to the scope and depth of their effects on the application as a whole. Rules, for example, may initiate a completely algorithmic feedback control loop, succeeding only when the control loop meets some convergence criterion.

An Example

An example of this algorithm/heuristic blend is presented in Panel 4, which illustrates part of an actual accelerator run in which the control system's start-up process was performed. The goal of this part of the start-up procedure is to reach a specified terminal voltage (energy potential for accelerating the particle beam). The terminal voltage is determined by the amount of electrostatic charge deposited from a moving belt onto the Van de Graaff generator's terminal, as determined by the belt charge selsyn: Increasing the belt charge increases the terminal voltage. Various factors, such as spark damage to the terminal and the current of the particle beam, alter the relationship between belt charge and terminal voltage, so the relationship is non-linear and subject to deviations.

As the expert system works its way through the start-up procedure, the stage is reached where the belt charge selsyn must be turned up until the required terminal voltage is exceeded slightly [17]. Although this *could* be implemented as a purely deterministic feedback loop, PACES instead encorporates inferencing into the control loop in an effort to detect any problems that might arise during start-up. This is analogous to operators who follow a general, well-established start-up procedure while simultaneously using their expertise to watch for unexpected problems (such as terminal voltage slew rate, poor vacuum, or improper SF_6 pressure). Hence, the algorithmic nature of the start-up procedure

is augmented by diagnostic heuristics to produce a smart, flexible and robust automated start-up process.

Panel 4 captures this start-up process as smoothed real-time accelerator data logged from the McMaster KN-3000. The graph plots the belt charge selsyn (thin line) and terminal voltage (thick line) as functions of time. It is important to note that the value shown for the selsyn is the *set point* that the host commanded the selsyn to turn to; the actual selsyn setting may require several seconds to reach this set point.

The control power is switched on at time 0, which produces a sharp rise in the terminal voltage to its uncharged baseline value (approximately 250 kV). The start-up process then proceeds to verify several safety interlocks (e.g. cooling water turned on, tank SF_6 pressure within acceptable limits). Next, the Van de Graaff drive motor is switched on, followed by the power supply for the belt charge. At approximately 18 s, the expert system begins turning up the belt charge selsyn. A fact stored in the knowledge base indicates that there is a "dead zone" in the selsyn, and the first 11 full turns of the selsyn will have no effect on the terminal voltage. The expert system therefore quickly turns the selsyn through 11 turns (requiring roughly 15 seconds).

The feedback control loop is entered at 35 s, at which time the controller gives the selsyn a slight increase, and waits until the rate of change of the terminal voltage reaches zero (indicating that the charging voltage has "caught up" with the control action). The belt charge increase continues in this step-like manner until the terminal voltage exceeds the specified value by no more than 10 kV. This goal is reached approximately 80 s after the start-up procedure was commenced. Once the specified terminal voltage is reached, the start-up procedure goes on to extract ions from the accelerator's ion source and focus these particles into a beam. The entire start-up process reaches its goal when the particle beam striking the first Faraday cup possesses the specified energy and current.

Fault Detection and Diagnosis

The most important duty of the PACES expert system is the timely detection of faults followed by expedient remedial action. This fault detection and diagnosis relies on heuristic inferencing because there is a distinct lack of sensors that could be used to directly detect the many types and locations of faults; rather than easily, repeatedly polling

hundreds of sensors, the control system must rely on inferring anomalous behaviour from the available telemetric data (derived from the control panel's meters). This is just the way that skilled operators diagnose problems: they are drawn to abnormal readings on the control panel meters, reason about causes and then try to rectify the problem.

PACES is able to classify faults into a continuum of severity (Figure 4). Low severity faults cause minor problems (such as slight beam instability) that can be ignored by the control system unless they become more severe. During such low severity conditions, the expert system will limit operation of the accelerator in certain ways, such as imposing an upper bound on the permissible terminal voltage and/or beam current. (For example, poor vaccum would "de-rate" the maximum terminal voltage to 1 MV rather than the normal 3 MV until the vacuum improves. This problem can arise after the accelerator has been opened to air for maintenance, and requires several hours after re-sealing to achieve optimal vacuum; during this time the machine is still operable, but not at maximum rating. Thus, the expert system is capable of "conditioning" the accelerator in a manner analogous to a seasoned operator's use of the machine under similar circumstances.)

High severity faults may require the control system to initiate safing (cut off of particle beam) or complete shut-down, or may cause a hardwired safety interlock to trip. In the latter case, the control system must recognize the trip and, if appropriate, attempt recovery.

The fault continuum is dominated by a wide range of severity that represents faults that are too severe to ignore but not severe enough to trigger a shut-down. It is for this wide variety of cases that the expert system is utilized to analyze the anomalous symptoms and produce a diagnosis. Due to the large number of possible faults, and the low number of indicators, the expert system is charged with finding the most plausible explanation for the anomaly. As an example of this diagnosis, consider the problem of "no beam" (Figure 5): The start-up procedure has progressed to the point at which, under

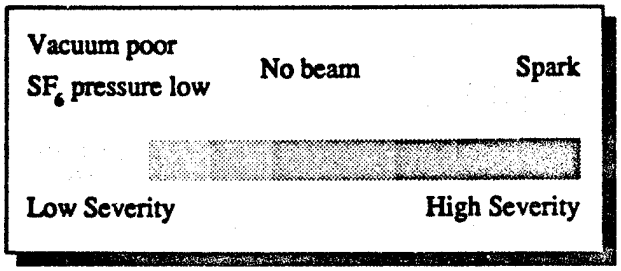


Figure 4. Severity of faults.

normal circumstances, a beam current should be measured at the first Faraday cup (Panel 1) [18].

The particles that form the accelerator's beam are the charged ions of a gas plasma. The gas is stored in a pressurized bottle [19]. Gas is let out of the bottle through a thermomechanical leak controlled by the *gas* selsyn: Turning up the gas selsyn heats up the thermomechanical leak, increasing the gas leakage rate. The gas flows into the near-vacuum *ionizing chamber* where a radio frequency (RF) oscillator ionizes the gas atoms. The *extraction* selsyn is used to control a negative voltage at the open end of the ionizing chamber which causes the positively charged ions to flow out of the chamber into the accelerating column.

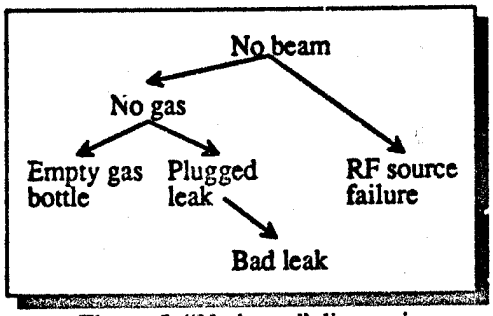


Figure 5. "No beam" diagnosis.

Given the above description, the "no beam" problem occurs when there is no measurable beam current on the Faraday cup even though the terminal voltage is correct and the gas and extraction selsyns have been adjusted properly. The lack of beam is most likely caused by either failure of the RF source (due, perhaps to a failed electronic component such as a vacuum tube), or a lack of gas to be ionized. To begin the diagnosis, the expert system consults the knowledge base to determine when the gas bottle was last changed. If the gas bottle has not been changed for some specified time, the expert system can suggest that the bottle needs to be re-filled. If the gas bottle was recently changed, the lack of beam is probably not caused by an empty bottle [20]. Given that the bottle is not empty, the expert system can hypothesize that the problem probably lies in the thermomechanical leak mechanism. This hypothesis can be automatically tested: The gas selsyn is zeroed, the gas source is switched to the other gas bottle, and the gas selsyn is turned back up. If this test succeeds in producing a particle beam, the expert system can conclude that the original gas bottle's leak mechanism is faulty. If the test fails, the expert system can infer that there is an electrical problem shared by *both* thermomechanical leaks (e.g. their power supply).

Although overly simplified, the "no beam" example illustrates the extent to which PACES uses knowledge-based inferencing to diagnose accelerator faults and resolve conflicting symptoms. In many situations, the expert system can be engineered to perform explorative troubleshooting (such as switching gas bottles) to reduce its search space during fault diagnosis. In some cases, the expert system is able to overcome the problem and continue accelerator operation; other problems result in shut-down and the posting of a diagnostic message that the operator can use to affect repairs.

Neural Network Controller

An interesting facet of the PACES control system is the implementation of neural networks on the SBC [21]. Data logged from many accelerator runs can be organized into training sets to train a neural network to recognize the characteristics of a stable particle beam. The PACES knowledge base can then include a repertoire of neural network "programs" (weight sets and neural topographies), and the expert system can then select a *template* to configure the SBC to perform rapid fine tuning of a certain type of particle beam. In this scenario, the neural network assumes control of the accelerator until the operator chooses to alter the beam or the beam's stability deviates beyond preset limits. When such alteration or deviation occurs, the SBC notifies the host computer and the expert system assumes control. After the beam is re-stabilized, the SBC may be re-configured with a new template, and resumes control.

Concluding Remarks

The Particle Accelerator Control Expert System is now beginning its third year of development. The first year saw the formulation of the control problem, the construction of the control circuitry and the evaluation of various expert system shells and programming environments. The second year centered around implementation of the user interface and knowledge engineering for the start-up and shut-down procedures. It is planned that the third year will concern broadening of the knowledge base to encapsulate a greater capability for fault detection and diagnosis, as well as the addition of the neural network controller. It is also hoped that the research project will be extended to cover control of the accelerator's beam line.

PACES is promising to be a useful tool for both expert and novice operators. Its automation of the repetitive aspects of accelerator operation relieves the operator of some

of the drudgery of accelerator control. The expert system's diagnostic facility, when completed, will serve to bridge the expertise gap between seasoned operators and new trainees. In this regard, the MS-DOS/Windows environment is a user-tunable GUI that can be tailored to different levels of expertise and to different personal tastes. Such interfaces are extremely valuable in enticing non-computer operators to embrace this new technology for machine control. Moreover, the dedicated and consistent approach to fault detection is believed to improve safety considerations [22] since the computer, unlike the human operator if busy at another task, does not need time to "plug in" to the current operating environment to make a key decision on shut-down.

Acknowledgements

The authors wish to thank Mr. Frank Strain and M. John Brisson, Defence Research Establishment Ottawa, for their generosity as sources of expert knowledge on the operation of the KN accelerator and for their kind assistance during the testing phase of the control system.

Additionally, Mr. Bob McNaught is recognized for his thoughtful insights in the faculty of expert consultant during design of the interface electronics.

Finally, the authors acknowledge the Department of National Defence for the funding support that has made this project possible.

Notes and References

1. Dept. of Electrical and Computer Engineering, McMaster University, 1280 Main St. W., Hamilton, ON, Canada, L8S 4K1. (416) 525-9140, Ext. 7262.
2. Applied Computersystems Group, Dept. of Computer Science and Systems, McMaster University, 1280 Main St. W., Hamilton, ON, Canada, L8S 4K1, (416) 525-9140, Ext. 2891.
3. McMaster Accelerator Lab, McMaster University, 1280 Main St. W., Hamilton, ON, Canada, L8S 4K1.
4. Scientific Authority. Dept. of National Defence, Defence Research Establishment Ottawa, Electronics Division, Nuclear Effects Section, Building 29, Shirley Bay, Ontario.
5. Manufactured by High Voltage Engineering Corp., Massachusetts, U.S.A.
6. Dept. of Computer Science and Systems, McMaster University.
7. PACES may also be used with the KN-4000 at the McMaster University McMaster Accelerator Lab and the KN-3000 at the Dept. of Physics, University of Guelph.
8. Lind, P.C., Poehlman, W.F.S., "Implementation Considerations for PACES: The KN-3000 Particle Accelerator Control Expert System", Proceedings of the Third Annual Symposium on Expert Systems in the Dept. of National Defence, Royal Military College, Kingston, Ontario, Canada, May 2-3, 1991.

9. The accelerator shown is the KN-3000 at the McMaster Accelerator Lab, McMaster University.
10. 8051-based Single Board Computer, HiTech Equipment Corp., San Diego, CA.
11. Microsoft Corp., Redmond, WA.
12. Borland International, Scotts Valley, CA.
13. Texas Instruments, Austin, TX.
14. Intellicorp, Inc., Mountain View, CA.
15. Windows performs non-preemptive round robin task switching.
16. As described in the *Neural Network Controller* section, the SBC is "programmed" by the expert system to perform limited scale fine tuning using a neural network.
17. During the later stage when the particle beam is actually produced, the terminal voltage tends to drop, so the machine is initially set to a terminal voltage higher than necessary.
18. The "no beam" problem can also arise during steady state beam maintenance when the beam is suddenly cut off.
19. The accelerator at McMaster is able to switch between two gas sources.
20. This alternative is still possible (i.e. the gas was left on high for a long period of time and the bottle emptied prematurely), but is less likely.
21. Lind, P.C., Poehlman, W.F.S., Stark, J.W., "Design of an Expert-System-based Real-time Control System for a Van de Graaf Particle Accelerator", paper accepted for publication in the Proceedings of International Conference on Artificial Intelligence in Engineering, University of Waterloo, Waterloo, Ontario, Canada, July 1992.
22. Private communication, Mike James, Senior Accelerator Licensing Officer, Atomic Energy Control Board, Ottawa, March 1991.

Bibliography

Armstrong, W.W., Gecsei, J., "Adaption Algorithms for Binary Tree Networks", IEEE Transactions on Systems, Man and Cybernetics, SMC-9, 1979, pp. 276-285.

Dwelly, A., "An Implementation of Adaptive Logic Networks", Unpublished technical report, Dept. of Computer Science, University of Alberta, 1990.

Harmon, P., Mans, R., Morrissey, W., Expert Systems: Tools and Applications, John Wiley and Sons, New York, 1988.

Hu, D., C/C++ for Expert Systems, Management Information Source, Portland, OR, 1989.

Ichikawa, Y., Sawa, T., "Neural Network Application for Direct Feedback Controllers", IEEE Transactions on Neural Networks, Vol. 3, No. 2, March 1992.

James, J.R., Suski, G.J., "Survey of Some Implementations of Knowledge-Based Systems for Real Time Control", Proceedings of 27th IEEE Conference on Decision and Control, Austin, TX, Dec. 7-9, 1988.

Johnson, C.D., Microprocessor-based Process Control, Prentice-Hall, NJ, 1984.

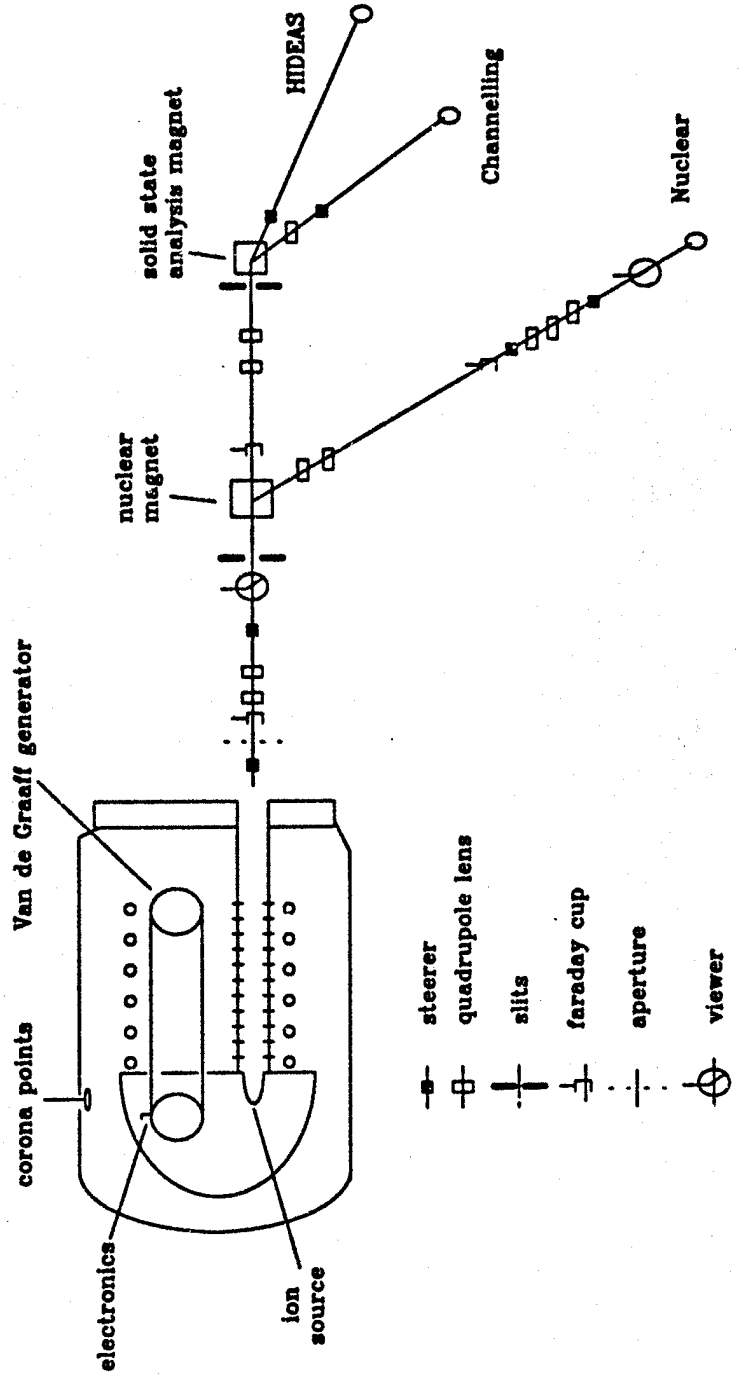
Parsave, K., Chignell, M., Khoshafian, S., Wong, H., Intelligent Databases: Object-oriented, Deductive Hypermedia Technologies, John Wiley and Sons, New York, 1989.

Partington, D., "Artificial Intelligence in Process Control", Measurement and Control, Vol. 21, No. 6, July-August 1988, pp. 177-178.

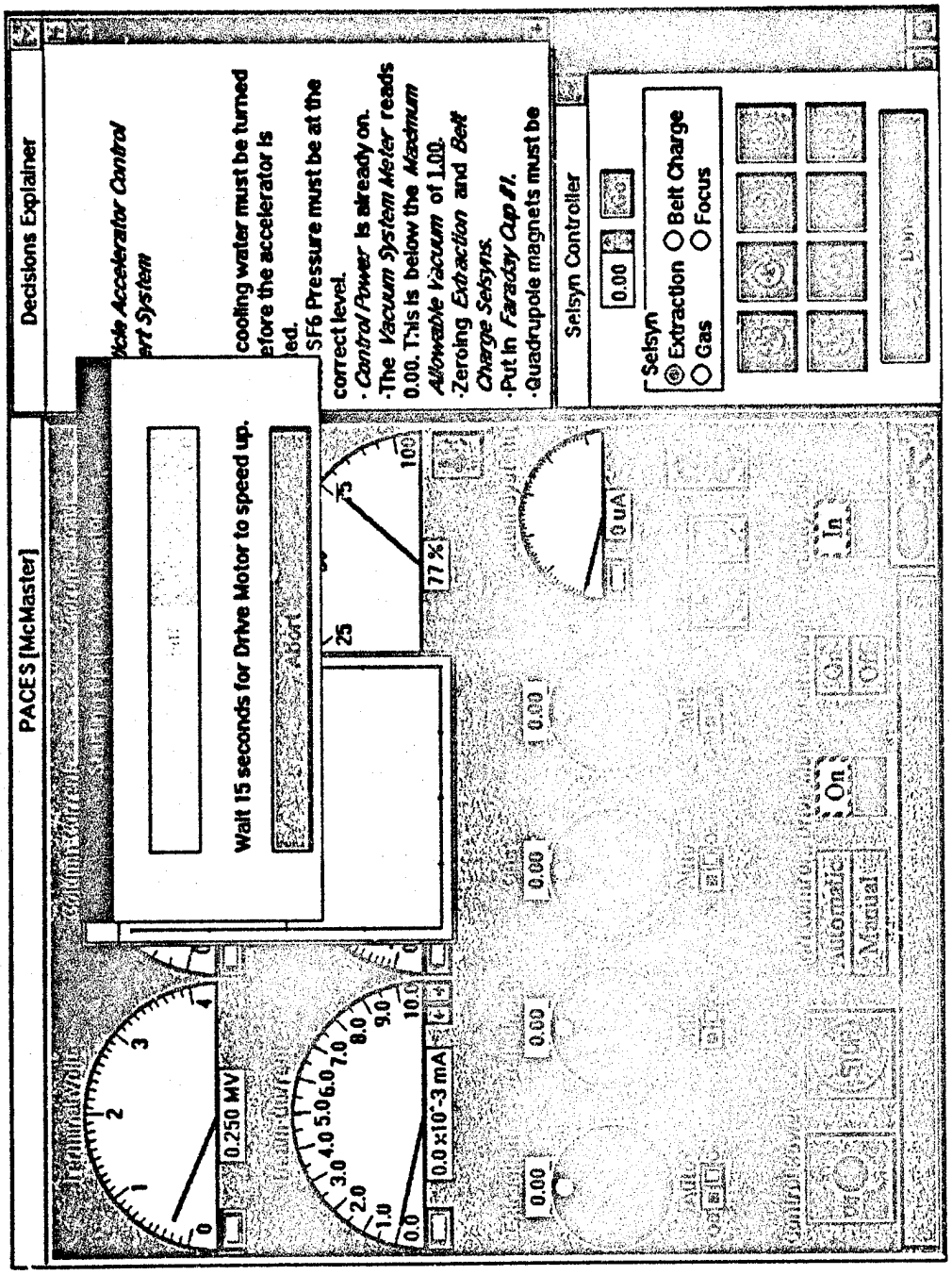
Silbar, R.R., Schultz, D.E., "Automation of Particle Accelerator Control", Proceedings of Computers in Engineering, San Francisco, CA, July 31-Aug. 4, 1988.

Tello, E.R., Object-Oriented Programming for Artificial Intelligence: A Guide to Tools and Systems Design, Addison Wesley, New York, 1989.

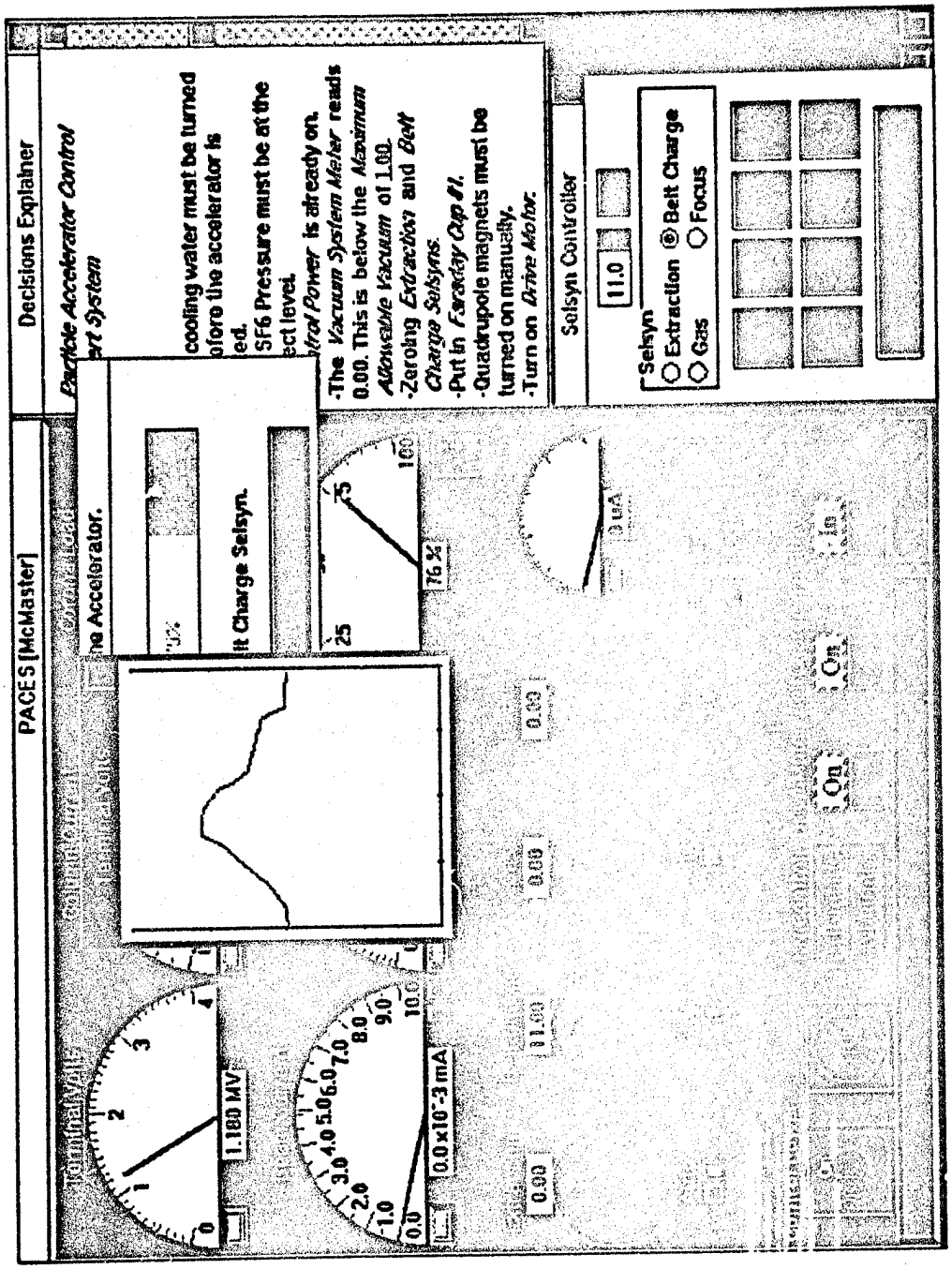
Zadeh, L., "The Calculus of Fuzzy If/Then Rules", AI Expert, Vol. 7, No. 3, March 1992.



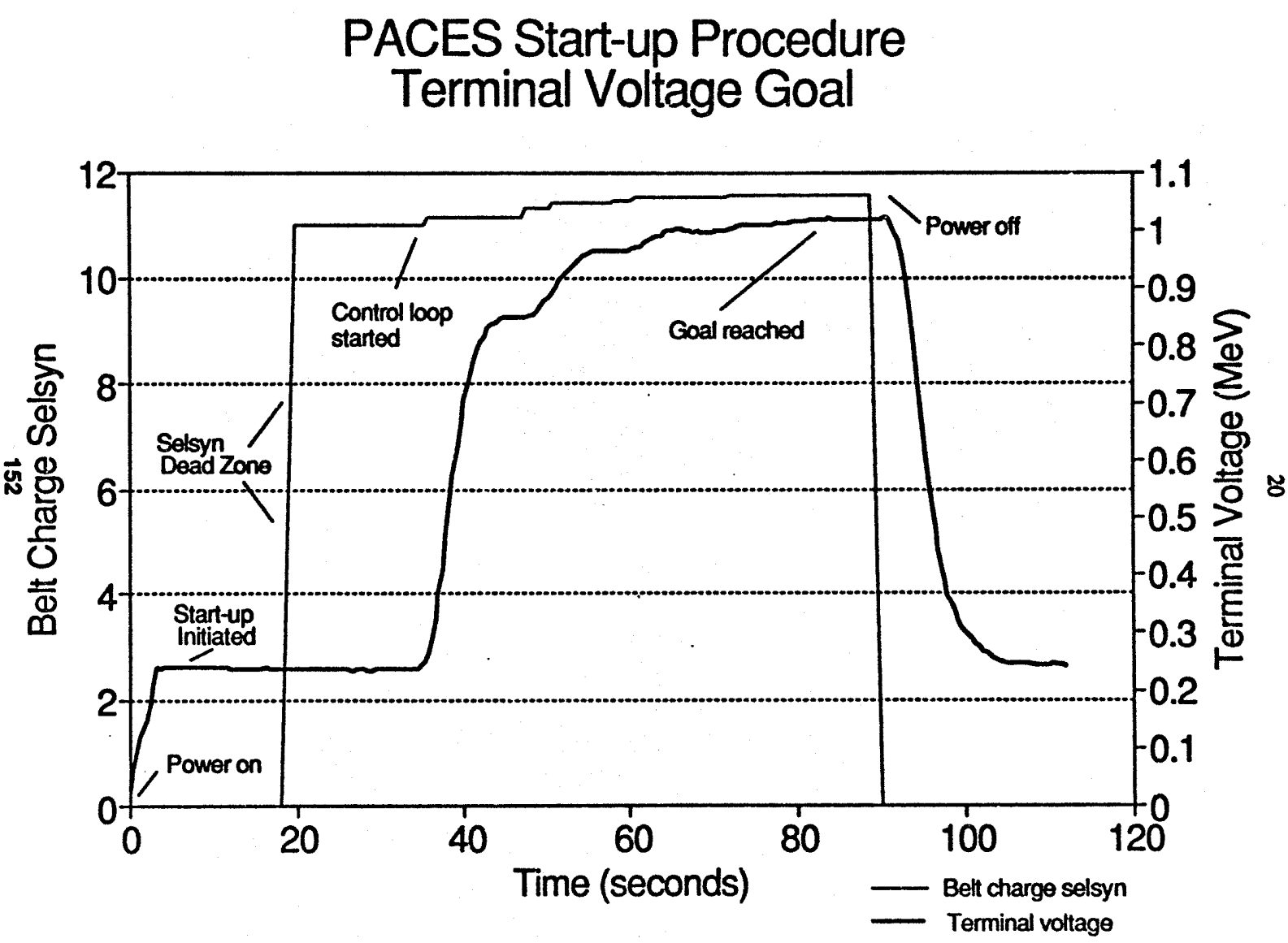
Panel 1. The McMaster KN-3000 Accelerator.



Panel 2. PACES user interface during Start-up Procedure.



Panel 3. Stripchart of Terminal Voltage during Start-up.



Panel 4. Terminal voltage goal of start-up procedure.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

THE HYPERMEDIA TECHNOLOGY AS A FRONT END TO KNOWLEDGE PROCESSING SYSTEMS

P.R. Roberge[1]

Abstract

Technological improvements can empower people, enabling them to accomplish tasks that once seemed impossible. Potent computer-based technologies are increasingly reshaping our expectations and hopefully serving human needs. At present the CF is facing a number of challenges such as mastering increasingly complex technology and systems while facing also increasingly severe shortages of recruits with above-average intelligence and static or shrinking budgets. The increased requirements for complex expertise in the CF has to be addressed by means independent of the constant drain of corporate memory and know-how imposed by posting and transfers.These challenges can be partly addressed by the development of artificial intelligence (AI) tools which would support some critical phases of the decision-making processes.

Two types of AI tools have recently gained the general confidence of both developers and users of these technologies and reached the main stream of computer goods. But these two technologies, database management systems (DBMSs) and expert systems (ESs), have had little interaction with each other. It is clear, however, that future decision-support systems will require both the problem-solving capabilities of ESs and the data-handling capabilities of DBMSs.This paper will describe how the hypertext or hypermedia technology is capable of integrating different knowledge processing technologies such as DBMSs and ESs while creating very flexible user interfaces for decision-support systems.

[1]Associate Professor, Dept of Chemistry & Chemical Engineering, RMC, Kingston

INTRODUCTION

Some of the most famous expert systems (ESs) have not gone beyond the development stage because they could not be timely integrated in the workplace, they could not communicate smoothly with users.

A major strength of ESs is that within a narrow domain they are able to capture much of the procedural expertise that a human expert uses to make decisions. Although ESs often fail to perform at the same level of competence as human experts, they can effectively serve as intelligent advisers for new-experts freshly posted to a new engineering position. An ES can thus guide a novice in making decisions which are consistent with those that a human expert would make.

Weaknesses of ESs include their inability to capture some of the important nuances of expert decision making [1]. ESs may typically offer accurate advises for general or routine problems, but are quite inefficient to meet unusual or exceptional cases. Even when the recommended course of action is adequate, the steps taken by ESs to arrive at a conclusion are not the same ones that a human expert would use. This makes it difficult for such systems to explain their behavior in a way that is useful to human users [2].

One of the main challenges faced by designers of such systems is the development of high grade user interfaces. Wide acceptance and use of any computer program presuppose a good interface and the task is not necessarily easy. It is common knowledge amidst workers in the field[3] that designing a good interface can consume up to one half of a project's development time. A few general observations have been made[4] on which to base design requirements for user interfaces:

- No single interface can satisfy all categories of users, nor can it meet all the needs of a single user over time
- Any interface should be amenable to custom-tailoring by the end user
- Interfaces should be easy to understand and designed to avoid information overload
- An interface should be transparent and never leave the user in the dark

The hypertext technology is ideally suited to accomplish these demanding goals and facilitate the design of intuitive user interfaces for decision-support systems. While ESs are designed to solve specific problems, hypertext systems are built to convey information. These two objectives are flip sides of the same coin. Some of the inefficiencies of ESs can be overcome by integrating the features of hypertext and some of

the weaknesses in hypertext systems can be overcome by using ESs. This complementarity will be illustrated with examples from systems recently developed to either the prototype level or simply as proof of concept for teaching or training applications. But first a brief introduction of both the hypertext and ES technologies will be presented as a preamble to the choice of tools made for these projects.

TOOLS

Hypertext Systems

While the term hypertext was coined by Ted Nelson during the 1960s[5], the concept itself can be traced back to Vannevar Bush's 1945 description of 'the memex'[6]. The first serious attempt to build a memex did not take place until 20 years after Bush's description. In 1968, Doug Engelbart, then at the Augmented Human Intellect Research Center (Stanford Research Institute), conducted a dramatic live demonstration of his Augment system during the Fall Joint Computer Conference where he worked collaboratively on a hypertext document with a colleague 500 miles away[7].

In the following years both interest and activity in hypertext have grown steadily. Some of the more important milestones have been described in the introductory paper of a special issue of Communications of the ACM[8] which was devoted to hypertext as a form of electronic documentation or information management. An excellent survey of existing hypertext systems with a critical review of their strengths and weaknesses was published in 1987[9]. In this article the operational advantages of hypertext were described to be the following:

- ease of tracing references
- ease of creating new references
- hierarchical or non-hierarchical information structuring
- global and local views can be mixed effectively
- multiple functions customized documents
- modularity and consistency of information
- task staking

There are many ways in which associative linking or non-linear writing and reading can be used. Today, you can find hypertext links in on-line help systems in a great

variety of systems such as Microsoft Windows™, personal information managers, text retrieval systems, ESs and application generators. But the programs that use hypertext fully are authoring software systems that are designed to create electronic manuals, training courses and other large text based applications. The two products that were selected in the present project to create an hypertext interface to some expertise management systems had been recently reviewed in a popular PC computer magazine[10].

The first system (HyperWriter™ from Ntergaid) was this magazine Editor's choice. It was said to have nearly every type of link available, from various kinds of text links to graphics, video and digitized sound. This system was also described to be extremely attractive for its support of runtime versions, both for their price (free) and their navigational tools such as the ability to call a graphics map of links in the document, see a reading history, tag files and leave bookmarks.

The second system (Guide™ by OWL International) was said to come close to the first in its functional richness[10]. This Windows™ application was believed to have superior formatting capabilities and a very good control over graphics. It also had the advantage of being nested in Windows which meant easy navigation between this software and other available Windows graphics and databases software.

Expert Systems

There has been, in the past two decades, a virtual explosion of interest in the field of ES development. ESs has quickly evolved from an academic notion into a proven and highly marketable product, one that offers an efficient and effective approach to the solution of real-world problems. By the end of the 1980s the implementation of at least 2000 ESs had already been documented within the corporate world alone[11].

ESs differ from conventional programming in the sense that it is a genuine approach to decision making or to a methodology in the support of decision making. In order to become an expert in rule-base development it is necessary to fully understand the multiple elements associated with such development:

- Knowledge acquisition
- Knowledge processing and inference
- Validation of the knowledge base
- Implementation and maintenance

Two commercial ES shells or tools were examined for their incorporation in

information processing systems. These have been described as middle range software packages[12].

Personal Consultant Plus(PC+) by Texas Instruments

The translation of technical know-how and common sense into programmed heuristics can be made by using such a commercial shell that synthesizes many years of research and development. The PC+™ development tool uses the same knowledge representation scheme which was used for the MYCIN and later the EMYCIN projects at Stanford University[13]. The primary knowledge representation scheme in PC+ is production (if/then) rules which describe an inference that can be made about a certain situation. This rule architecture has some advantages[14]:

- Humans can relate to production rules as "rules of thumb"
- Rules are very modular and therefore easy to modify
- Rule specification is non-procedural and can be arranged by the system itself
- Rules allow for explanation of reasoning

The PC+ development environment contains all the features necessary to create and test knowledge-bases (simple rule entry language, editor, listing facility and debug aids). PC+ is written in LISP which makes it relatively slow to operate. One characteristic of this shell is its rigid user interface which depends entirely on pop-up menus.

Nexpert Object by Neuron Data Inc.

Nexpert Object 1.1™ is an innovative ES shell that combines rules and objects very efficiently. Nexpert is written in C and fully utilizes the facilities offered by running in the screen oriented Microsoft Windows™ environment. One of the key features of this system is a common rule format for forward and backward chaining. Using such a common format for both types of reasoning signifies that the knowledge-base can be used in multiple ways to derive a maximum of information from a given set of instructions. Nexpert Object runtime platform interrogates the user by means of custom-made forms displayed on the screen which supports pop-up and horizontal bar menus. But a serious disadvantage of this system is the high price one has to pay for these run-time platforms.

EXAMPLES

Four recent projects will be briefly described to demonstrate how the hypertext

technology was used as a front-end to knowledge base systems. The first two systems were developed to the prototype level in order to serve as demonstrators or proof of concepts. The intended users were to be newly posted personnel requiring specialized training as well as practicing engineers and scientists requiring some degree of expert support. The second pair of systems presented as examples are only in their early design stage. The intention, at the moment, is to introduced these systems as lecture material or used them as course support in some of RMC undergraduate courses on Chemical and Materials Engineering.

Power Sources Information System

The first example presented is a software prototype which was developed for the Director Avionics, Simulators and Photography (DASP) during the accomplishment of a Task[15] aimed at defining an approach for the development of a flexible knowledge-base system for the management of chemical power source systems. The first prototype itself was entirely programmed in PC+ in a multi-tier structure. At each search level the user was given the opportunity to select one of the choices displayed on the screen. This selection was made by using cursor arrows to highlight a choice and <CR> to enter the choice itself. Fig. 1 illustrates a typical consultation screen of this first prototype.

The prototype developed with PC+ was found to be too rigid, insufficiently user-friendly and monotonous. While the integration of graphics in the system was theoretically possible, its effects were found to be quite unpredictable and degraded extensively the operational reliability of the system itself. Such a drawback was considered to be serious since a fair portion of the information required to build the prototype referred to battery design features which are better represented by drawings. The PC+ cooperation with commercial database systems was not very satisfactory since it could not override the production of the series of title screens produced each time the systems were called forward.

The attempt to transfer the knowledge-base, developed during the first part of this project, to a different ES programming environment (Nexpert Object) also turned out to be disappointing even if the system tested was said to be graphic oriented and user friendly. Unlike PC+, Nexpert Object lets the developer control the user interface but, even then, its form-creating facilities were found to be inadequate for the task and the user interface remained relatively monotonous(Fig. 2). Although it was possible to incorporate graphics in the development version this remained impossible for the runtime versions themselves.

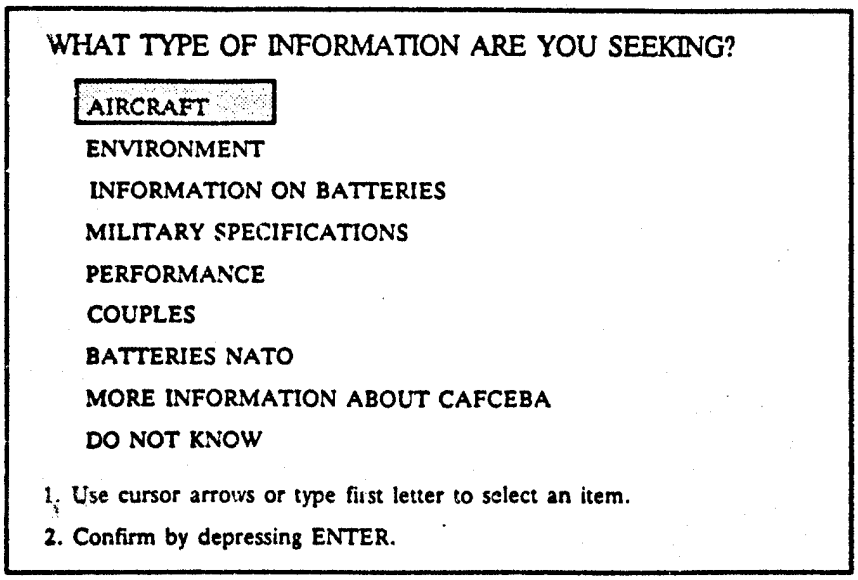


Fig. 1 Typical consultation screen of the PC+ chemical power sources knowledge-base prototype.

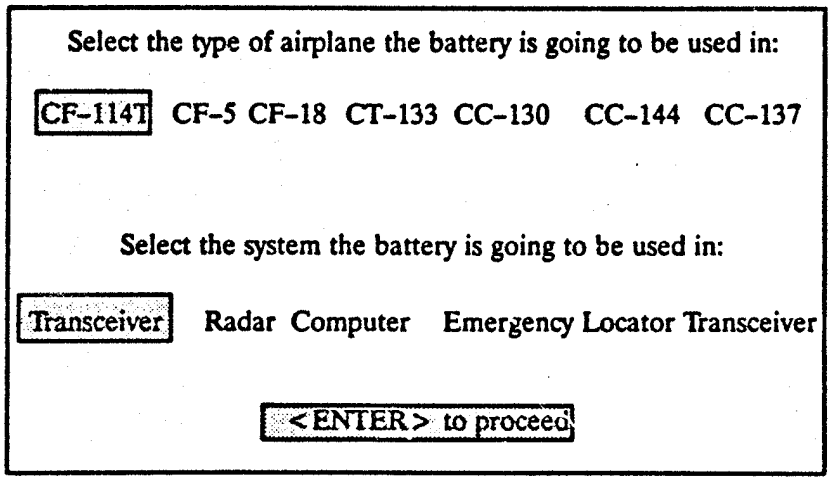


Fig. 2 Typical consultation screen of the Nexpert Object chemical power sources knowledge-base prototype.

The latest prototype was built in the hypertext environment (HyperWriter) as an interlinked network of nodes (Fig. 3) The system contains various types of battery

information as texts or graphics and can be triggered to access database files (Paradox 3.5™ by Borland) by means of C coded routines or fire the ES which had been developed in Nexpert.

The hypertext environment was also successfully tested during this project for its ability to incorporate external graphic files. Several pictures were scanned and added to the system with overall links or by a screen sensitization method ('hot spots').

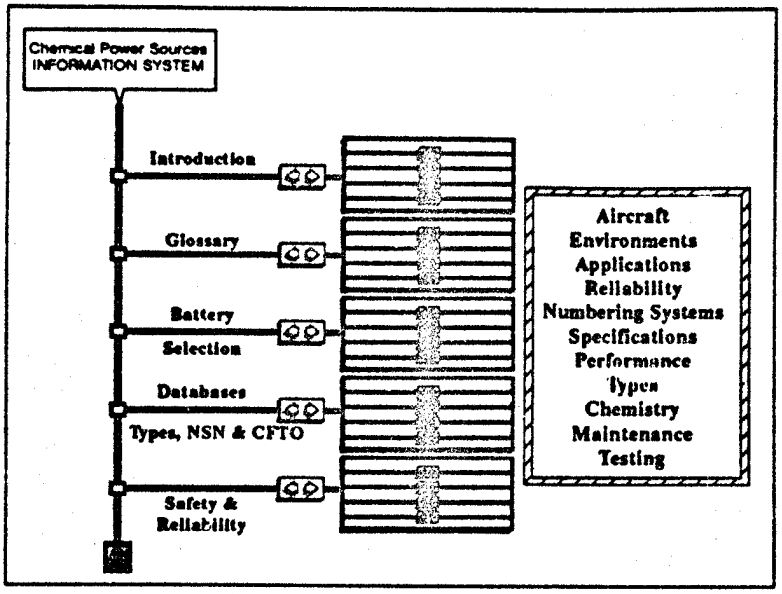


Fig. 3 General structure of the hypertext battery information system with links to databases and an ES prototype for the selection of batteries.

Failure Analyst

The second prototype presented as an example of a hypermedia front end to ESs was developed during the completion of a Master thesis in Chemical and Materials Engineering[16]. The general goal of Failure Analyst was to support personnel involved with material problems where Stress Corrosion Cracking (SCC) is either suspected or a potential cause of failures.

The architecture of Failure Analyst followed the recommendations outlined by Turban[17]. The top goal or objective, which was here to determine the extent to which SCC can be considered to be a serious problem, was first established by prioritizing all contributing factors to failure. An inference tree (Fig. 4) helped to organize these factors in

an orderly and comprehensive manner. This inference tree also served as the backbone structure to construct the various programs that can be activated during a single consultation. Several test cases were selected to test the effectiveness of Failure Analyst to come to plausible conclusions and the results of these tests were compared to answers provided by human experts in an effort to validate the computerized approach.

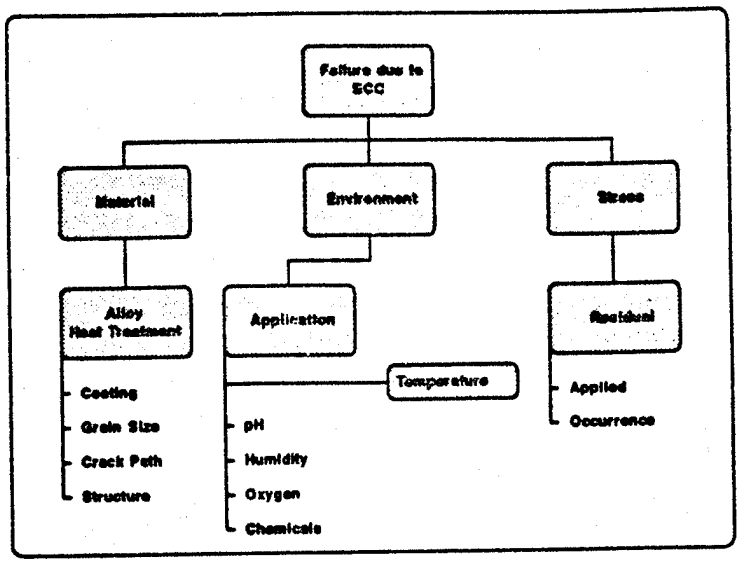


Fig. 4 Inference tree of SCC factors considered by Failure Analyst.

While the early knowledge organization of Failure Analyst was conceived around the PC+ platform, its construction was directly made in the hypertext environment (HyperWriter). The prototype fully used the non-linear information processing capabilities of the hypertext environment by constructing queries as series of cards or windows interlinked in a very structured fashion. With such an organization a continuous control was kept over the user progress through the maze of possible questions and interconnected non-sequential links. When programming in a hypertext environment it is important to achieve a balance between the depth of information sought and ease of navigation without getting lost in hyperspace.

Failure Analyst was constructed by assigning help or explanation windows to each question with the possibility of always looping back to the initial question of a series.

Such a loop prevented the user to avoid answering important questions while providing timely information. Over 300 links and 100 windows were assembled in the prototype user interface. The data transfer to nested programs (written in QuickBASIC by Microsoft) was achieved with scripted commands. The interactive links permitted by these commands became an essential component of the organization of the whole system itself. Fig. 5 describes the overall architecture of Failure Analyst.

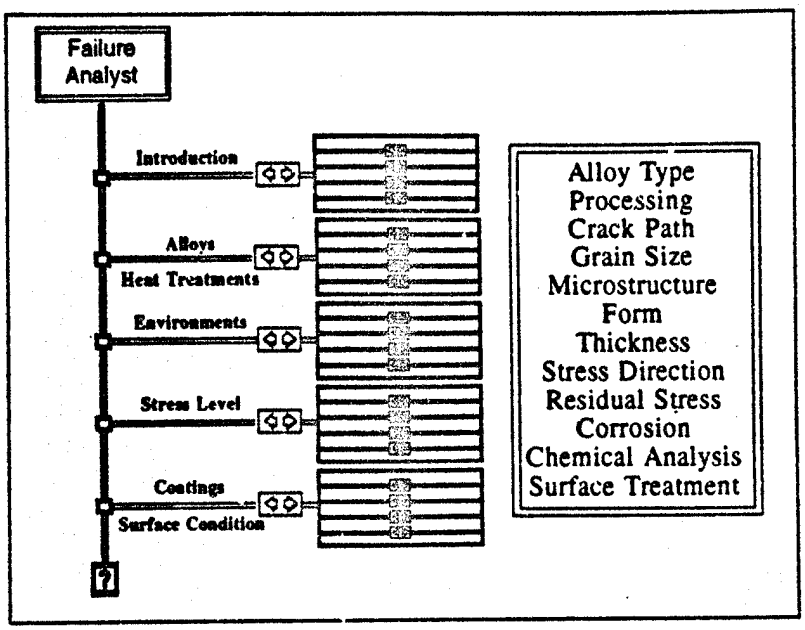


Fig. 5 General structure of Failure Analyst with links to important components of the knowledge base.

Corrosion Prevention by Coatings

Putting a barrier between a corrosive environment and the material to be protected is a fundamental method of corrosion control. It is, in fact, the most widely used method of protecting steel and other metals. The concept of adding coatings to surfaces is so ancient that its beginnings are lost in the mists of history. Mechanisms by which coatings protect can be summarized as follows[1ε]:

- They prevent contact between the environment and the substrate
- They restrict or limit contact between the environment and the substrate as with most

organic coatings

- They release substances which are protective or which inhibit attack
- They produce a protective electric current

A common reason why protective coatings do not perform well is often because they have not been considered as systems. Most successful coating engineers approach a coatings project in much the same way that they approach any other engineering problem, beginning with the design of surfaces to be protected and ending with schedules for monitoring the completed work.

A global representation of the features which would be included in this hypertutorial is presented in Fig. 6. The generic information on coatings which will be used for this prototype already exist in a textbook form [19]. The system will make full use of the programming easiness provided for this kind of projects by the Windows environment. Tables will be constructed using the Excel database system which communicate to the hypertext environment (Guide) through transparent Dynamic Data Exchange (DDE) links. The coatings tutorial will also be linked to a fully operative commercial ES specialized in the selection of coatings (Coating Counsel by Counselware Inc. [20]). Coating Counsel is designed to select optimal coating systems for industrial maintenance applications. It can write complete Construction Specification Institute format specifications in a matter of minutes.

Information Processing for Materials Performance

This system under consideration would help newly posted engineers to learn the complexity of maintaining systems integrity. Such a task has become increasingly more complex since modern plants are also increasingly automated, larger and more expensive to own and operate. The high costs of maintenance resources and equipment availability have thus become a critical factor in companies profitability. The parallel growth in sophistication and technological content of maintenance work has made its management far more difficult[21]. In order to improve maintenance management and scheduling, it is imperative to consider introducing in normal operations some of the modern tools that have been developed for accurate surveillance and monitoring of critical systems.

In recent years a tremendous progress has been made for the understanding of effects caused by various types of material damage. Very complex mechanisms are gradually being defined for stress corrosion cracking, hydrogen attack, hot corrosion and

other forms of deterioration. For practical structural integrity applications proper decisions about the type of inspection and its frequency have to be made.

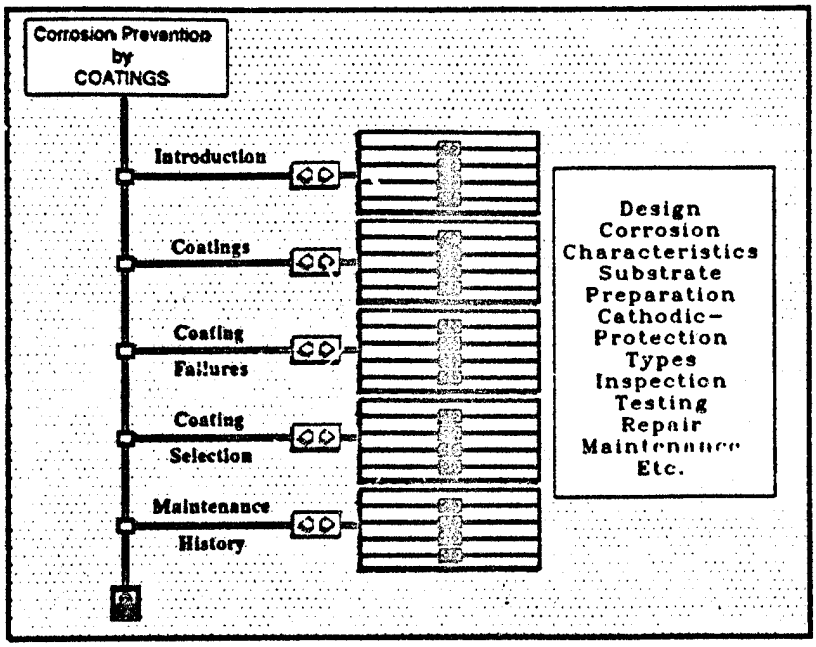


Fig. 6 General structure of the coating hyper-instructor with links to knowledge components and to a fully operative commercial coating selector ES (Coating Counsel).

Several inspection types can be used and the choice of a specific scenario will depend on the accuracy and cost of the inspection. The accuracy of a given technique must be sufficient to detect defects considerably smaller than those which could result in failure. A cheaper and less accurate technique used frequently could be equivalent costwise to a more expensive and more accurate technique used less frequently leading, however, to greater reliability. Regardless of the technique that has been chosen, the critical decision remains with the frequency of application. This decision depends on three things[22]:

- the extent of damage that might remain invisible to the technique
- the rate of damage occurrence with time
- the extent of damage the structure can tolerate

The applicability of a given inspection technique to a particular problem will heavily depend on the nature of the problem itself. The internal inspection of a pipe system

could, for example, be attempted by using a crawler device which would carry an optical probe and corrosion sensors[23]. But such a practice can only be performed during shut-downs. The inspection of in-service systems is also often complicated by the fact that these systems operate at relatively high temperature in a closed mode. While traditional techniques are effective in detecting and sizing existing defects there are a number of disadvantages associated with their use[24]:

- traditional inspection technique can only be used on high temperature process equipment when the equipment is out of service.
- inspection is generally limited to suspected problem areas due to time, accessibility and cost factors.
- the fundamental lack of knowledge on the effect of exposure of given materials to specific environments.

Computers are ideally suited to keep track of all the different deterioration rates for thousand of pieces of plant equipment. But it is important that the inspection files be arranged or organized for an efficient management analysis. A few systems are being commercialized to provide these services. MAPS™[25], for example, includes programs for information management integrated with other software systems that handle erosion - corrosion surveys, plant inspections, OSHA compliance surveys and other specialized tools that can handle Computer Assisted Design (CAD) data bases or simply run problem-solving routines. Another example of maintenance enhancement through integrated software is the MASC™[26] Inspection Program which claims to be able to assist inspection engineers in the management of effective inspection, predictive maintenance, reliability engineering and corrosion control programs. Table 1 lists some operations provided by such an integrated inspection tool.

The present goal for this project is to create a platform with would help to clarify which sectors of this vast domain could be feasible targets for very focused ES given the expertise available. The Windows environment seems presently the environment of choice to develop an open ended architecture for a preliminary prototype. The easy access to information processing software in this environment will make the creation of a modular structure interestingly possible. In its preliminary version the system will be used to teach the fundamentals of corrosion monitoring showing particularly how the information gathered during off-line and on-line inspections can be translated into plant integrity information understandable by management personnel.

Table 1 Typical modules of a complete integrated system for inspection and maintenance planning.

Inspection		Communication
Data	Records	
• general information • engineering information • operational conditions • materials	• observations • recommendations • measurements • corrosion rate • retirement thickness • life to retirement • inspection interval • next inspection • due date	• memos • work requests • maintenance inquiries • operational inquiries • engineering inquiries • materials inquiries • inspection approvals • shutdown reports • budget preparation

CONCLUSION

The weaknesses found in traditional ES shells for the construction of user-friendly and stand-alone ESs can be greatly overcome by creating high grade user interfaces with commercial hypertext or hypermedia systems. The increased flexibility gained by using this combination of tools can be felt during the initial design of such systems. The examples presented in this paper illustrate how it is possible to consolidate some area of expertise by using an open ended architecture to organize an ES development strategy. Complex systems can be designed efficiently and simple prototypes rapidly developed to illustrate the global approach of a knowledge processing project. The goal of stabilizing an area of expertise can be demonstrated and implemented while keeping users interested in systems under development.

REFERENCES

1. Alty, J.L., "The Limitations of rule-based expert systems," in Knowledge-Based Expert Systems in Industry, Tiri Kriz (Ed), 1987, pp. 17-22.
2. Stenton, S.P., "Dialog management for co-operative knowledge based systems," The Knowledge Engineering Review, Vol. 2, No. 2, 1987, pp. 99-122.
3. Bobrow, D.G.,Mittal, S. and Stefilk, M.J. "Expert Systems: Perils and promises," Communications of the ACM, Vol. 29, No. 9, 1986, pp. 880-894.

4. Barsalou, T, Chavez, R.M. and Wiederhold, G. "Hypertext interfaces for decision-support systems: A case study," Medinfo 89, Proceedings of the sixth Conference on Medical Information, 1989, pp. 126-130.
5. Nelson, T.H., "Getting it out of our systems," in Information Retrieval: A Critical Review, G. Schechter(Ed.), Thompson Books, Washington, DC, 1967, pp. 191-210.
6. Bush, V., "As we may think, "Atlantic Monthly, Vol. 176, No. 1 (July 1945), pp. 101-108.
7. Engelbart, D.C. and English, W.K., "A Research center for augmenting human intellect," in Proceedings of the 1968 Fall Joint Computer Conference, Montvale, NJ, AFIPS Press (Fall 1968), pp. 395-410.
8. Smith, J.B. and Weiss, S.F., "Hypertext," Communications of the ACM, Vol. 31, No. 7 (1988), pp.816-819.
9. Conklin, J., "Hypertext: An introduction and survey," IEEE Computer, Vol.2, No. 9 (Sept 1987), pp. 17-41.
10. Fersko-Weiss, H., "3-D Reading with the hypertext edge," PC Magazine, May 28 (1991), pp. 241-282.
11. Ignizio, J.P., "Introduction to Expert Systems," Mc=Graw-Hill Inc., New-York, NY, 1991.
12. Harmon, P., Maus, R. and Morrissey, W., "Expert Systems: Tools and Applications," John Wiley & Sons Inc., New York, 1987, 287 p.
13. Buchanan, B.G.and Shortliffe, E.H., "Rule-based Expert Systems: The MYCIN experiments of the Stanford Heuristic Programmning project," Addison-Wesley, Reading, MA, 1984.
14. Turpin, W.M. "Texas Instruments-Personal Consultant Expert System Developming Tools," in Proceedings of First annual Conference on AI and advanced computer technology, April 1985, pp. 167-171.
15. "Research in the Area of Battery Expert Systems," DND Development Project D6002 02174, Ottawa, Oct 1989.
16. Bryson, G. "Failure Analyst: An Expert System for Stress Corrosion Cracking," Royal Military College of Canada, Kingston, 1992.
17. Turban, E, "Decision Support and Expert Systems," Macmillan Publishing Co., New York, 1990, 495 p.
18. Brasunas, A. de S., "Corrosion Basics: An Introduction," National Association of Corrosion Engineers, Houston, Texas, 1984,355 p.

19. Munger, C.G., "Corrosion Prevention by Protective Coatings," National Association of Corrosion Engineers, Houston, Texas, 1984, 522 p.
20. Murray, S., "Coating Counsel: Automated Preparation of Painting Systems Specifications," Counselware, Montreal, Quebec, 1991, 10 p.
21. Kelly, A. and Harris, M.J., "Management of Industrial Maintenance," Newness-Butterworths Managemant Library, New York, 1979, 356 p.
22. Conley, M.J., Angelsen, S. and Williams, D., "A modern approach to equipment integrity solutions," CORROSION/91, paper no. 169, National Association of Corrosion Engineers, Houston, TX, 1991.
23. Carpenter, J., Stanton, D. and Kumar, A., "Corrosion inspection by a robotic crawler," CORROSION/91, paper no. 563, National Association of Corrosion Engineers, Houston, TX, 1991.
24. Farraro, T.D. and Allevato, C., "Applications of Acoustic Emission Testing (AET) for detection of defects in refinery process equipment during thermal gradients," CORROSION/91, paper no. 311, National Association of Corrosion Engineers, Houston, TX, 1991.
25. "Management Analysis and Productivity System (MAPS)," USI Technical Services Inc., Houston, TX, 1991.
26 "MASC," Kurtz and Steel Ltd., Mississauga, Ontario, 1991.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

A PROBLEM-SOLVING APPROACH TO KNOWLEDGE ACQUISITION

Capt J.G. Dumais[1] and Dr A.L. Jenkins[2]

Abstract

Current approaches to knowledge acquisition (KA) are primarily prototyping and modelling. Many expert system (ES) projects developed under both approaches never result in a working system, because of deficiencies in the ES development process. This paper defines a KA methodology which should solve some of these problems and consequently make ESs more affordable. A phased development process is proposed. The methodology uses the characteristics of the task to determine first the problem-solving (PS) model which best resembles the strategy used by the expert in solving problems. Furthermore, this PS model suggests the relevant knowledge types as well as the general structure in which they must be represented to be useful in solving problems. Based on this information, the methodology suggests appropriate techniques to further assist the knowledge engineer in the iterative elicitation of the relevant knowledge types. Finally, the elicited knowledge is validated.

[1]Graduate Student, [2]Professor, Dept of Engineering Management, RMC, Kingston

PART I LITERATURE REVIEW

INTRODUCTION

The fact that ES projects have been overshooting time and cost estimates at an alarming rate is not new [1]. Although project management practices are, in part, responsible for this situation, developers agree that the difficulties inherent in KA play a more significant role [2]. In short, KA is the most critical phase of the ES development process where knowledge from a variety of sources, and more specifically human experts, is elicited, analyzed and represented explicitly in a format which subsequently allows it to be coded for computer use. This paper is an attempt to define a KA methodology which should solve some of the problems associated with this process and consequently make ES development more affordable. This paper is divided into two parts, Part I gives a brief review of the literature on the subject of KA methodologies, and Part II presents, based on this review, the proposed KA methodology.

KNOWLEDGE ACQUISITION DEFINED

The terms knowledge engineering (KE), KA and knowledge elicitation are used somewhat loosely in the literature. Since a clear definition of these terms is fundamental to this paper, they are defined at this time. KE is that branch of the software industry concerned with the building of ESs [1]. KE is concerned with all aspects of system development, including analysis, design, implementation, and maintenance. KA is the process of eliciting, analyzing, and representing the knowledge used by human experts to solve problems in a domain [3]. KA is the most critical part of the KE process and occurs mostly during the analysis phase of the development. The definition of the term "methodology" is also important here. A methodology, in a software development context, is a systematic and planned approach for completing one or more of the phases of the development process [4].

Since KA is an integral part of KE, a short discussion of KE approaches is presented to place both in the proper context. Researchers generally distinguish between two KE approaches [5], prototyping and modelling, as shown in Fig. 1. Prototyping owes its popularity among knowledge engineers to the difficulties associated with KA. In prototyping, a simplified version of the ES is produced early in the development process and used as a focusing device for further KA cycles. This prototype evolves and in many cases becomes the final system.

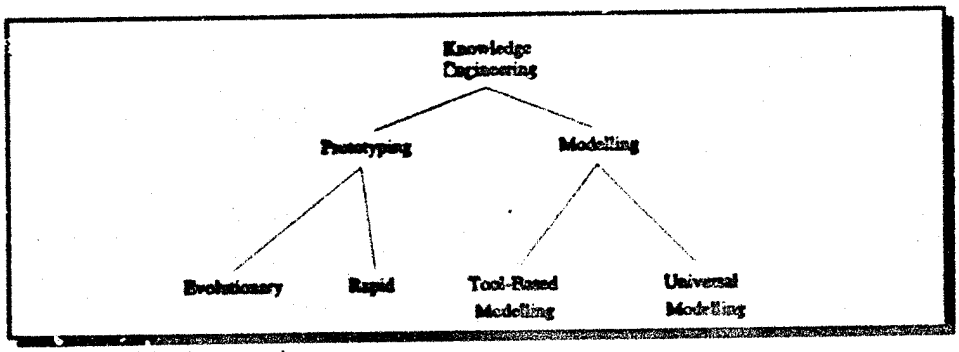


Figure 1. KE Approaches

The second KE approach views the ES development process as a modelling activity [6]. Modelling methodologies assume the existence of a limited set of repetitive problem-solving (PS) methods, which can be modelled and re-used across domains. The KA process under this approach consists of analyzing the expert's task, identifying a suitable PS method for this task, and using the structure provided by this PS method to structure and guide the top-down refinement of the knowledge base (KB).

KNOWLEDGE ACQUISITION METHODOLOGIES

Two KA methodologies are described in more detail in the following paragraphs to better illustrate the KE approach they support.

A Prototyping KA methodology

One of the first attempts at documenting a prototyping KA methodology is presented by Grover [7]. The methodology includes three phases: domain definition, fundamental knowledge formulation, and basal knowledge consolidation. In the initial domain definition, the knowledge engineer develops an understanding for the expert's language, determines the level of performance of the proposed system, and generates scenarios or case problems which are used to elicit the expert's task knowledge. In fundamental knowledge formulation the knowledge engineer proceeds with the elicitation of task knowledge, i.e. the elicitation of procedural knowledge from the expert using the case problems previously identified in domain definition. In the third phase, the system's reasoning strategy and its control knowledge are improved iteratively using the prototype. The prototyping process is shown in Fig. 2.

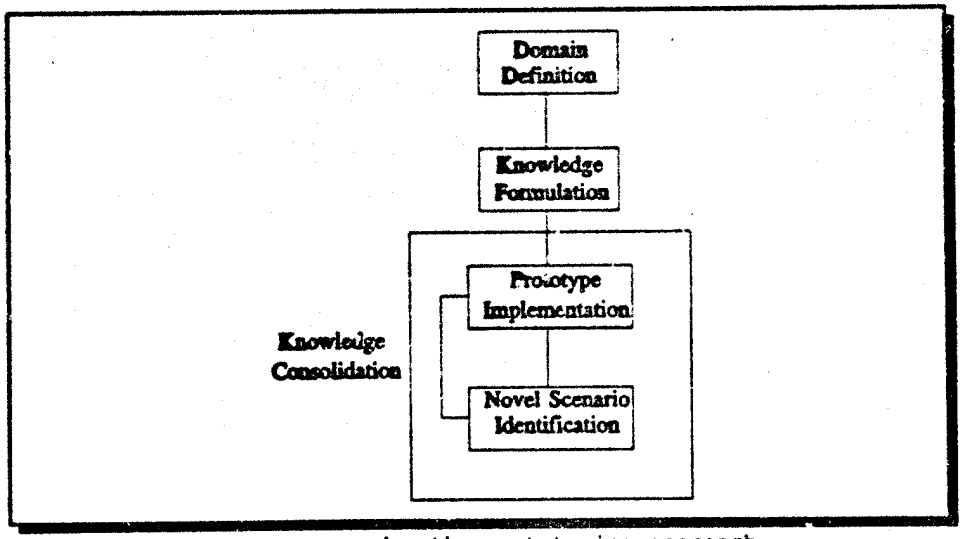


Figure 2. The KA process under the prototyping approach.

Prototyping is usually done using an ES shell (ESS). The selection of this shell is done early in the development process and based on the general nature of the expert task. A shell essentially provides an implementation formalism, such as rules, and a default search strategy such as backward chaining. The main advantage of using a shell is that the prototype development time is greatly reduced.

A Modelling KA Methodology

KADS [6] or Knowledge Acquisition Documentation and Structuring is an European-funded project developed through academic and commercial cooperation. The development phases under KADS are analysis, design, implementation, testing, and maintenance. KADS, unlike prototyping, emphasizes a clear distinction between the analysis, design, and implementation phases of its development process. At the heart of KADS lies the interpretation model (IM), which is a model for a PS method for a class of tasks. The four layers of knowledge used in KADS and how these layers are related to the components of interpretation and conceptual models are shown in Fig. 3. The four layers together constitute a conceptual model, whereas removing the domain layer, which contains all domain knowledge, yields an IM. So far, IMs are available for classification, heuristic and systematic diagnosis, suitability assessment, monitoring, prediction, modification, design, and planning [6]. It is these IMs that allow for the top-down refinement of the KB.

The analysis phase of KADS is divided into three cycles: orientation, problem identification and problem analysis. During orientation, the knowledge engineer acquires the expert's vocabulary and analyzes the task to select or construct an IM. Problem identification involves determining the conceptual structure of the domain and a thorough functional analysis of the task. Finally, problem analysis deals with users and operational issues which impact on the architecture of the prospective ES. The outputs of the analysis phase, which include at least a conceptual model and any

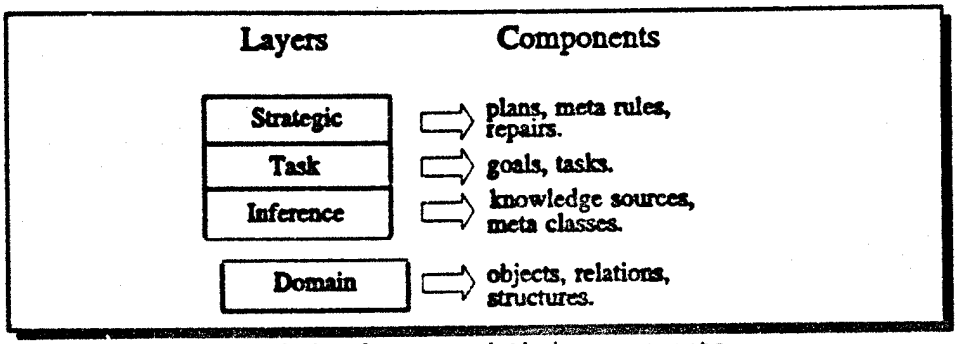


Figure 3. KADS' Knowledge layers and their components.

external requirements are input to a design phase in which these objects are transformed in the specification of the architecture of the ES [6]. Fig. 4 shows the three KA cycles and how they relate to the KA process under the modelling approach.

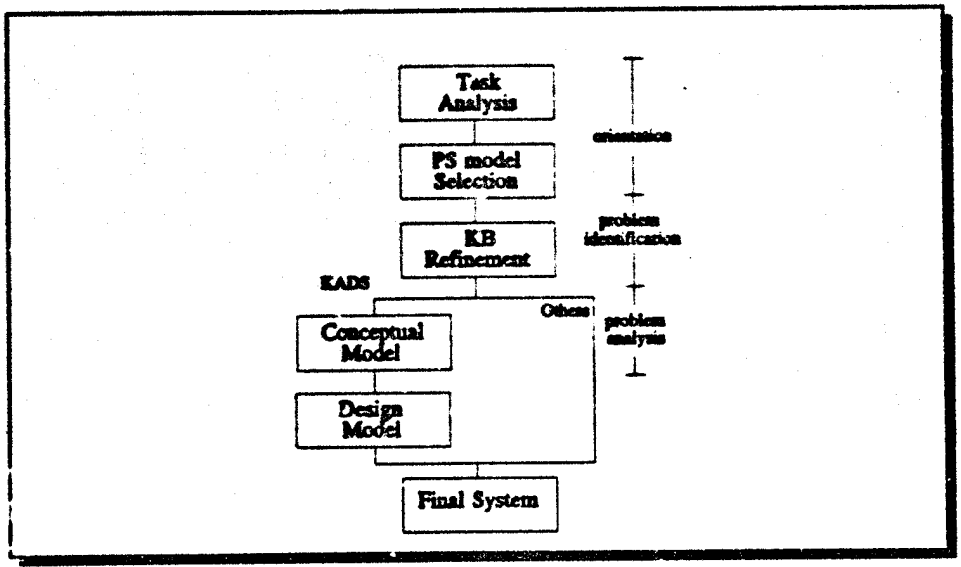


Figure 4. KA Process under the Modelling Approach.

7

CRITIQUE OF THE KA METHODOLOGIES

Based on the KA methodologies described above, a critique of both prototyping and modelling is now presented. The main advantages usually associated with a prototyping methodology are:

1) provides for speedy results which can be used to sell the project to an organization [8],
2) puts a functioning system into the hands of users quickly [9], and
3) encourages the active participation of domain experts and potential users [9],

However, these advantages are offset by a number of disadvantages which can be summed up by: prototyping encourages dangerous shortcuts through the development life cycle of a system [4]. More specifically, these disadvantages are:

1) encourages a return to the code, implement and repair life cycle, or poor software engineering practices [10],
2) leads to premature commitment to an implementation formalism which may be ill-suited for the domain and the task [10],
3) leads to the uncontrolled growth of the KB usually associated with a potential loss of structure [10],
4) encourages organizations to accept the prototype as the final system although, in most cases, it is not appropriate [11], and
6) does not address several design issues including: integration, documentation, reliability, maintainability, and security [12].

These disadvantages are further compounded in practice by the use of ESSs, which only offer a limited combination of implementation formalisms and inference strategies [6]. The modelling methodologies have, to some extent, attempted to address prototyping's main disadvantages. The

advantages usually associated with modelling are:

1) delays the commitment to an implementation formalism, therefore allowing the analysis to proceed without having to deal with design and implementation issues [6],
2) encourages good software engineering practices [6],
3) facilitates the management of the development process by providing a phased approach with milestones for sign-offs [6],
4) facilitates the KA process by allowing it to be model driven [6], and
5) guides KA by identifying the knowledge types relevant to the task at hand [13].

Based on these advantages, it appears that modelling is the more promising direction in which to proceed with KA. However, modelling is not without problems, which remain to be addressed to make this approach workable in practice. The problems to be addressed are 1) the techniques provided to the knowledge engineer to conduct task analysis are limited, 2) the mapping between the result of the task analysis and a taxonomy of PS models is only possible for a limited number of tasks, 3) the suggestion of appropriate elicitation techniques to acquire the relevant knowledge types, and 4) the better utilization of the expert in the overall KA process. The KA methodology proposed in this paper attempts to address these problems.

PART II A PROBLEM-SOLVING APPROACH TO KA

Part II presents the development of the proposed KA methodology. Although the development of this methodology is central to this paper, a KA methodology cannot be developed in isolation from the KE approach it supports. Moreover, the concept of an expert system development life cycle (ESDLC) also supersedes that of a KE approach. Consequently, the first section of Part II addresses the ESDLC, and presents a discussion of the

fundamental assumptions on which the KA methodology is based. The second section covers the most important stages in the analysis, where the bulk of KA takes place.

THE EXPERT SYSTEM DEVELOPMENT LIFE CYCLE (ESDLC)

A Phased Development Process

A phased approach characterizes the development of most commercial products, so why should ESs be the exception? The definition for ES reveals that, for the most part, they are similar to conventional computer programs. Therefore, it is logical to attempt to adapt the conventional System Development Life Cycle (SDLC) to derive a development life cycle suited to ESs. Moreover, the peculiarities of ESs do not affect the phases of the SDLC, but only the stages into which these phases are divided. For instance, ESs, like conventional programs, still require analysis, design, implementation, testing and maintenance. However, the stages into which analysis is decomposed are different for ESs. Fig. 5 shows the phases of the ESDLC and an exploded view of the analysis phase.

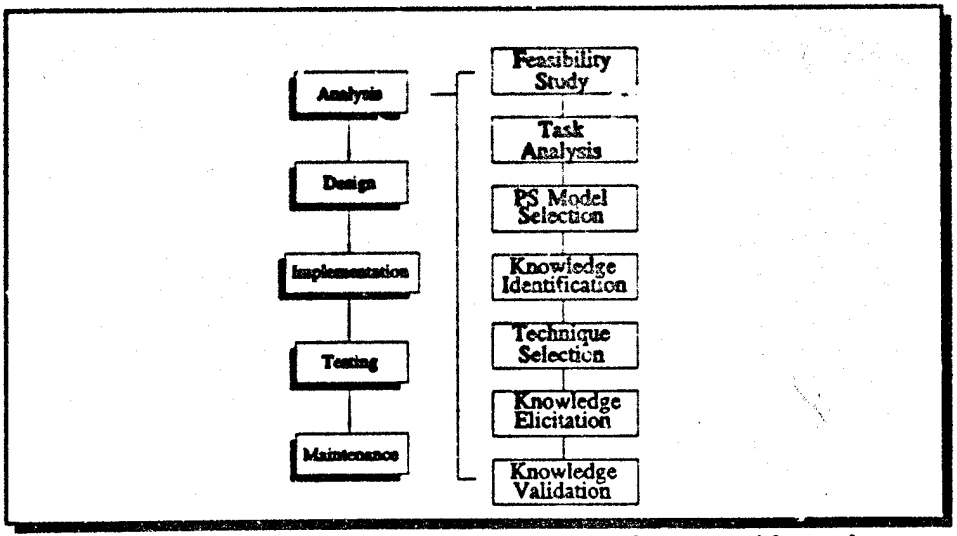


Figure 5. The Phases of the Expert System Development Life Cycle.

A phased development has the following main advantages 1) allows for the better management of the development effort by providing clear milestones, and 2) encourages a clear separation between its various phases. For example, analysis can proceed without concern for design and implementation issues. In a KA context, this decoupling is highly desirable since it allows the knowledge engineer to concentrate all his efforts towards the analysis of the expert's knowledge. Such a decoupling is, however, only possible if knowledge can be represented in an implementation-independent formalism during the analysis phase. An implementation-independent formalism is a notation used as a conceptual aid in synthesizing knowledge from the expert [14]. The implementation-independent formalism selected for this KA methodology is Systemic Grammar Networks or SGN, first used by Johnson and Johnson [14]. In short, a SGN is a graph composed of nodes and links. The nodes have names invented by the knowledge engineer but suggested by the data. The links are of four types, namely co-occurring, mutually exclusive, recursive, and conditional. A SGN is drawn in a later section to illustrate the notation.

Facilitating KA through Problem-Solving Models

Facilitating KA is most likely the key to ensuring a wider utilization of the ES technology in the industry. To this end, modelling appears the more promising direction in which to proceed. Consequently, the KA methodology proposed here follows this trend. For instance, the methodology supposes the existence of a limited number of problem-solving methods, which can be used to structure and guide the elicitation process. On the other hand, the methodology does not depend on a KA tool for the acquisition of knowledge from the expert. Both the knowledge engineer and the expert play a vital role in this KA methodology. The next section discusses the analysis phase of the ESDLC in more detail.

THE ANALYSIS PHASE

The analysis phase of the proposed KA methodology includes a feasibility study, task analysis, PS model selection, knowledge types identification, technique selection, knowledge elicitation, and knowledge validation as shown in Fig. 5. Each of these stages is now discussed in some detail. It should also be noted that emphasis is placed where the proposed KA methodology differs the most from previous proposals.

Task Analysis

Following commitment to the project, the aim of the task analysis is to identify the PS model most applicable to the expert's task. To this end, a taxonomy of task types is a useful tool because it provides a structured framework which can be used with consistency in the classification of tasks. The taxonomy shown in Fig. 6 is based on Clancey's [15], but has been altered in two ways. The first alteration involves the *modify* task. Whereas Clancey regards it as a sub-type of *assemble*, *modify* is defined here as a combination of *identify*, *design* and *assemble* and therefore *modify* is not included in our taxonomy. The second alteration concerns the addition of *classify*, which categorizes a system based on observable features, and *repair*, which prescribes remedies for identified discrepancies, both as sub-types of *identify*.

The task analysis is accomplished in an interview setting between the knowledge engineer and a domain expert. Using the above taxonomy to guide the process, the knowledge engineer first classifies the task as either analysis or synthesis. The task characteristics required for this initial classification are that analysis generally involves identifying sets of objects based on their features, and that synthesis usually requires that a solution be built from sub-problem solutions. To sum up, this classification procedure is repeated until all of the relevant task characteristics have been made explicit and used to traverse the tree shown

in Fig. 6. A task is classified when a terminal node is reached.

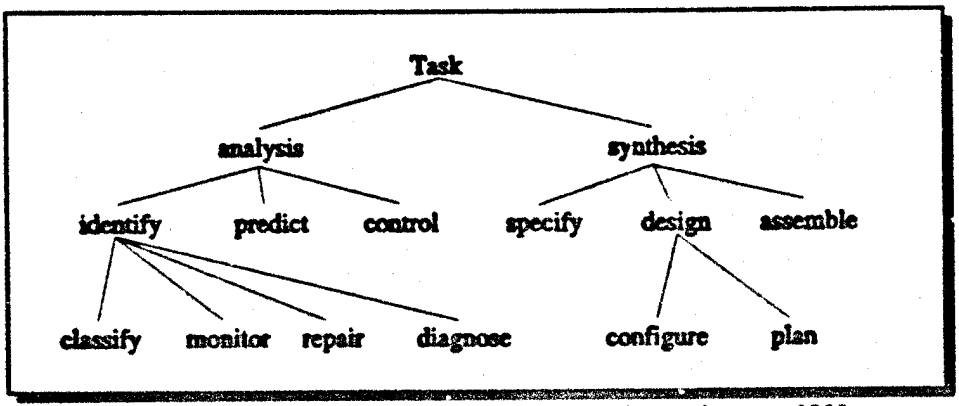


Figure 6. A Taxonomy of Task Types (Adapted from Clancey 1985).

Problem-Solving (PS) Model Selection and Knowledge Types Identification

Once a task type has been identified, the knowledge engineer proceeds to determine which PS model can be used to perform a task of this type. However, the methodology does not yet support a direct mapping between task types and PS models, for the simple reason that a taxonomy of PS models which completely covers the above taxonomy of task types does not yet exist. However, research in this area is continuing and should some day allow such a mapping. Regardless, the proposed KA methodology can still be described using the PS models that do exist for the *diagnose* or *configure* tasks. For illustrative purpose, the diagnosis PS model is selected. It should also be noted that the KA methodology is based on the PS model description proprsed by McDermott [16].

Cover and differentiate is the PS model associated with the diagnosis task. This PS model is suitable for tasks that have the following characteristics:

1) an identifiable set of symptoms each of which must be explained or covered
2) an exhaustive set of candidate explanations that cover these

symptoms can be pre-enumerated,

3) information is available which helps differentiate the candidate explanations for each symptom, and

4) usually only one candidate explanation is applicable for any given symptom.

The description of the tasks suitable to this PS model highlights the requirements for these knowledge types: 1) symptoms, 2) explanations or possible causes for these symptoms, and 3) differentiating knowledge or information that helps differentiate these symptoms. This information facilitates KA by providing the knowledge engineer with an indication of the knowledge which is relevant to the expert's task. Additionally, providing the knowledge engineer with the right tools for eliciting this relevant knowledge should ensure that the KA bottleneck is effectively removed or at the very least widened.

Elicitation and Analysis Techniques

The KA methodology proposed here is supported exclusively by manual elicitation techniques at this time. Although the trend is towards automated KA, it is believed that greater efficiency, especially in terms of expert utilization, can be achieved through the use of appropriate manual elicitation techniques and alternate knowledge sources. Another important principle in this KA methodology is that the use made of a piece of knowledge affects considerably the format in which it is required. The methodology does not ignore the existence of a domain or static structure but places more importance on the task or dynamic structure of the knowledge. Moreover, it is also believed that the format in which the knowledge is required directly affects the techniques which should be used for its elicitation. In other words, for each PS model specific knowledge types are required which necessitate the use of different knowledge elicitation techniques. Further detail on the elicitation techniques of this methodology are now presented using the diagnosis task and PS model

selected previously.

The first knowledge type required by the cover and differentiate PS model is a set of symptoms. Under current KA methodologies, the most likely method for obtaining this information is to ask the expert to provide a list of such symptoms. However, this is probably not the most efficient way to proceed, since a large number of symptoms may be involved. Moreover, this is a rather tedious task on which the expert's time is essentially wasted since no specific expertise is required to perform this activity. In this context, the proposed KA methodology turns to three alternate sources for this knowledge. Two of these sources are essentially documents while the other is not. A set of symptoms can be compiled by:

1) querying an existing information system (IS) regarding symptoms reported by users or customers, and
2) using the troubleshooting manual for a piece of equipment or system.

In the event that these two knowledge sources are not available, the required information can be elicited from non-expert diagnosticians in the domain using questionnaires, which are discussed further below. As stated above, expertise is not a prerequisite for all stages of the KA process. These three alternate sources all have the advantage of not requiring the active participation of the expert in the elicitation of an initial set of symptoms. The expert participation is, in fact, limited to the validation of this set, a task which is a lot less time consuming. Moreover, it is also possible that these alternate sources may produce a more complete set of symptoms. In the case of the documents, the number and range of symptoms elicited can easily surpassed that of one expert. A situation which can also be duplicated by a small number of non-expert diagnosticians.

In cases where the expert must be involved, the methodology suggests the utilization of indirect as opposed to direct elicitation techniques.

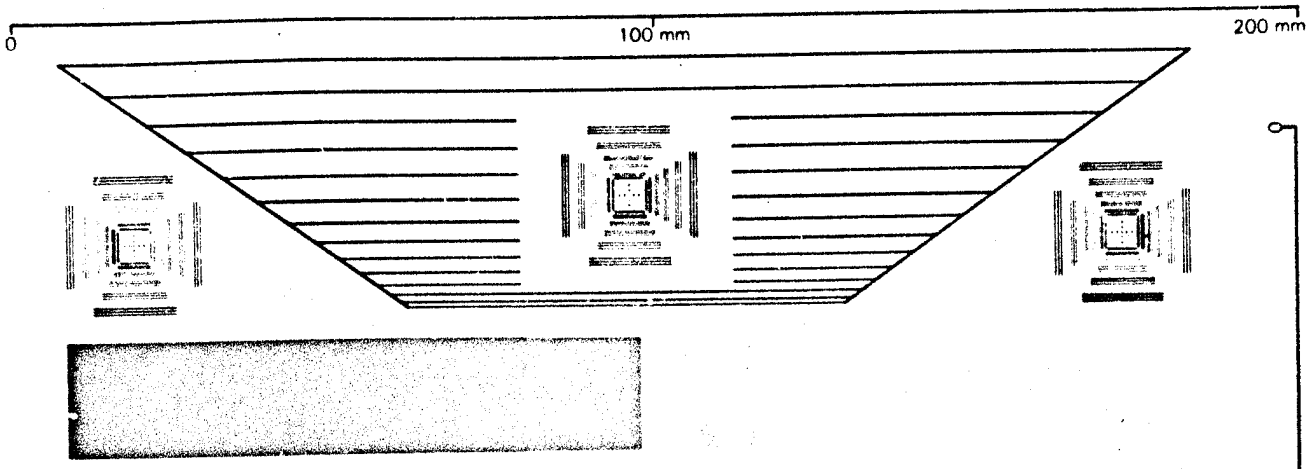
Two such techniques suitable to the elicitation of this knowledge are 1) questionnaires, which are generally good instruments to uncover objects in a domain [17], and 2) an unstructured interview. Questionnaires should be considered first because, like the previous techniques, they can be distributed to more than one expert or non-expert and therefore result in a better initial set of symptoms. Additionally, they can be filled out in a leisurely and relaxed atmosphere, i.e. away from the expert's place of work. The unstructured interview is the only method proposed by this KA methodology when dealing directly with an expert. The interview process is facilitated in this case by using either the system or a model of the system as a focusing device for the elicitation. For example, if a diagnostic system for a car engine is being built then the engine itself or at least a drawing representing that engine should be used in the elicitation of an initial set of symptoms.

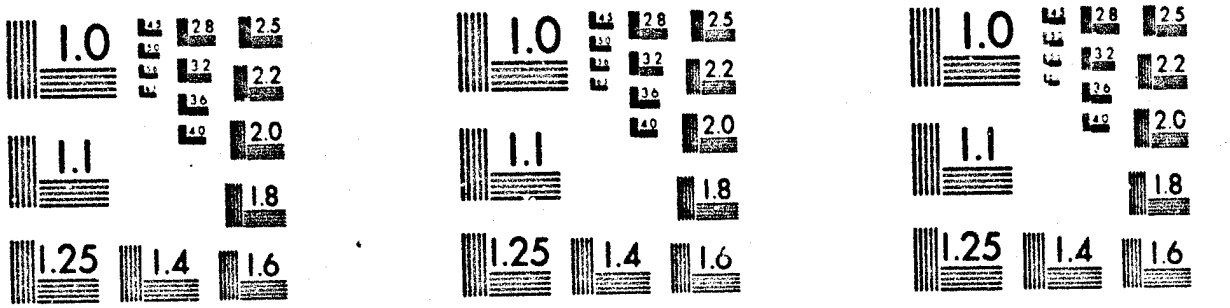
Covering knowledge, which explains each symptom, can be elicited in the same general manner as an initial set of symptoms. Caution should be used, however, that identified malfunctions are, in fact, related to a given symptom. This is especially important, for example, when querying an existing IS for symptoms and corresponding maintenance actions. These maintenance actions may, in fact, have little to do with the reported symptoms and the danger exists that unsubstantiated relationships will be established between symptoms and malfunctions. The KA methodology also suggests three additional elicitation techniques at his stage. These are dividing the domain [18], where the expert proceeds from observable symptoms to a malfunction, pure reclassification [18], where the process is from the malfunction to its symptoms, and finally, systematic symptoms to malfunction links [1], where a list of malfunctions and symptoms is used to determine the relationships that exist between the malfunctions and the symptoms. In this last technique, a list of malfunctions must first be built. Consequently, the technique should only be used after the first two, in order to gain a better understanding of the structure of the domain. It should be noted that all these techniques take advantage of the fact that

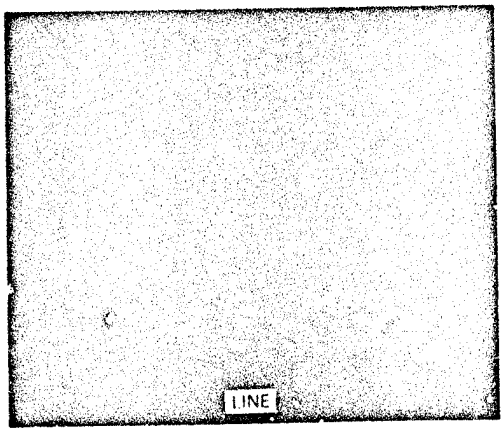
CONTROL TEST TARGET

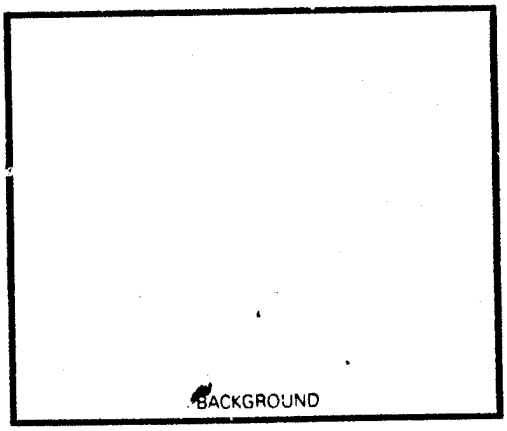
Kodak Quality Monitoring Program

2.4x









Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 5

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz 1 2 3 4 5 6 7 8 9 1

100 mm

200 mm

Sensitized Imaging Products

Business Imaging Systems Division

A-2289

7-87

the initial set of symptoms can be used to drive the elicitation of explanations for them. This, in turn, highlights the importance that this initial set of symptoms be as complete and as valid as possible. The acquisition of an initial set of symptoms and of the malfunctions explaining these symptoms provides the essential structure of the system under diagnosis as shown on the partial SGN in Fig. 7.

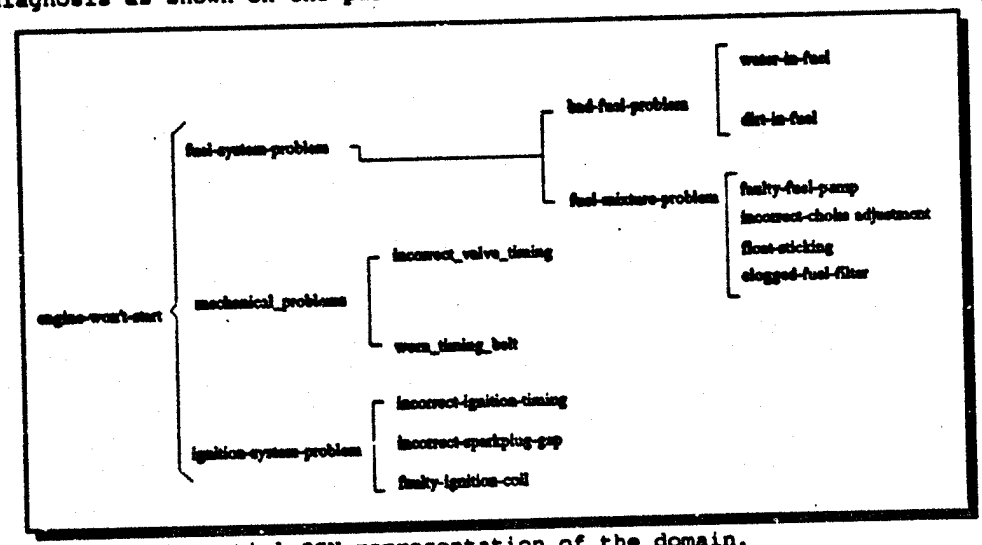


Figure 7. A partial SGN representation of the domain.

Differentiating knowledge is still required in order to complete the knowledge requirements of the cover and differentiate PS model. This type of knowledge is not easily elicited from the expert, because of the difficulty that the expert has specifying precisely how the information provided helps the differentiating process. Differentiating knowledge helps the knowledge engineer decide which of two explanations is the correct one for a symptom. Therefore, it seems that this type of knowledge is, in fact, related more directly to the task than the domain, or is more procedural than declarative. Consequently, the KA methodology proposes that the elicitation of differentiating knowledge be conducted using procedural knowledge elicitation techniques. The elicitation techniques proposed by the methodology are protocol analysis, goal decomposition [1], and Retrospective Case Description [18]. Protocols should be tried first

because they are most likely the most efficient technique to acquire procedural knowledge. This KA methodology stresses the utilization of concurrent as opposed to retrospective protocols, in which there is a danger of post hoc rationalization. Post hoc rationalization occurs when the expert in an effort to explain his reasoning embellishes his description with what he thinks should have been done instead of what was done. More specifically, two protocol techniques are suggested and should be experimented with. These are the technique of familiar tasks and the technique of tough cases [19]. It should be noted that not all experts respond to protocol techniques. In fact, some training may be required before the expert feels at ease with the procedure. If for some reasons protocols cannot be used, the other two techniques listed above should be tried, starting with Retrospective Case Description. A summary of the elicitation techniques proposed by the KA methodology is shown in Table 1. Above all, the knowledge engineer should be flexible, and other elicitation techniques should be sought if the expert is not receptive to the ones listed in Table 1. The selection of other techniques should, however, be based on the fundamental principle which the methodology stresses, that is the efficient utilization of the expert during the KA process.

Task Type: Diagnosis PS Model: Cover and Differentiate	
Knowledge Types	**Elicitation Techniques**
Symptoms	1. IS Query 2. Troubleshooting Guide Review 3. Questionnaires to non experts and experts 4. Unstructured Interview
Malfunctions	1. Troubleshooting Guide Review 2. Questionnaires to non experts and experts 3. Dividing the domain 4. Pure Reclassification 5. Systematic symptom to malfunction links
Differentiating Knowledge	1. Concurrent Protocols 2. Retrospective Case Description 3. Goal Decomposition

Table 1. Knowledge Elicitation Techniques proposed by the KA methodology

Knowledge Elicitation

So far, the proposed KA methodology has addressed all of the stages leading to knowledge elicitation. In doing so, the methodology has emphasized structure and guidance as the key to success in KA. Knowledge elicitation marks a turning point in this respect for the methodology. Indeed, the methodology holds that one cannot bluntly ask an expert to describe his expertise and domain knowledge and expect a coherent and organized answer. To the contrary, knowledge elicitation remains, due to the very nature of knowledge, an iterative process, where the knowledge engineer and the expert actively interact to arrive at a correct representation of the expertise. The elicitation process requires a number of sessions and, in this respect, the methodology does not prescribe a specific plan to follow, except maybe for the one suggested by the knowledge types relevant to the PS model which is described briefly in the previous section. This means that for the diagnosis domain, the knowledge engineer would proceed with the elicitation of an initial set of symptoms, followed by the elicitation of explanations for these symptoms. Finally, differentiating knowledge would be elicited from the expert.

Knowledge Validation

The final analysis stage before committing to design is one of the most important because errors made early in the analysis are usually costly to fix. Knowledge validation in the proposed methodology is facilitated by two factors. First, knowledge elicitation is conducted iteratively, which means that both the knowledge engineer and the expert use the mediating representation extensively during knowledge elicitation with the result that the elicited knowledge is, to some extent, validated through the elicitation process. Second, a mediating representation, unlike a prototype, is free from implementation details, and therefore much closer to the expert's representation. This, in turn, greatly facilitates knowledge validation.

The validation techniques suggested here also support the efficient utilization of the expert, a fundamental principle for this methodology. For example, cases which range from the most to the least common are used to evaluate the completeness and the accuracy of the elicited knowledge. Knowledge validation should also be done with non-experts and especially prospective system users. Besides actual cases, the mediating representation can, in conjunction with questionnaires, be used for validation purpose, for the reasons given above.

CONCLUSION

A KA methodology based on PS has been presented. The methodology uses the characteristics of the task to determine first the PS model which best resembles the strategy used by the expert in solving problems in his domain. Furthermore, this PS model suggests the relevant knowledge types as well as the general structure in which they must be represented to be useful in solving problems. Based on this information, the methodology suggests appropriate techniques to further assist the knowledge engineer in the iterative elicitation of the relevant knowledge types. Finally, the elicited knowledge is validated.

REFERENCES

1. Greenwell, M., Knowledge Engineering for Expert Systems, Ellis Horwood Ltd, Chichester, U.K., 1988, 184 p.

2. Freiling, M., Alexander, J., Messick, S., Rehfuss, S., and Shulman, S., "Starting a Knowledge Engineering Project: a step-by-step approach.", AI Magazine Vol. 6, No. 3, 1985, pp. 150-165.

3. Waterman, D.A, A Guide to Expert Systems, Addison-Wesley. Reading, MA., 1986, p. 392.

4. Whitten, J.L, Bentley L.D, and Barlow, V.M, Systems Analysis and Design Methods, 2nd ed., Irwin, Boston, MA., 1989, pp. 415-416.

5. Vob, A., "Model based Knowledge Acquisition.", Proceedings of the first Workshop on Information Systems and Artificial Intelligence: Integration Aspects, Springer-Verlag, Berlin, Germany, 1990, pp. 256-272.

6. Breuker, J. and Wielinga, B., "Models of Expertise in Knowledge Acquisition.", in Topics in Expert System Design, eds. G. Guida, and C. Tasso, North Holland, Amsterdam, 1989, pp. 265-295.

7. Grover, M.D, "A pragmatic Knowledge acquisition Methodology", Proceedings of the 8th International Joint Conference on Artificial Intelligence, 1983, pp. 436-438.

8. Rolston, D.W, Principles of Artificial Intelligence and Expert Systems Development, McGraw-Hill, New York, NY, 1988, pp. 146-148.

9. Martin, J. and Oxman, S., Building Expert Systems: A Tutorial, Prentice Hall, Englewood Cliffs, NJ., 1988, pp. 168-171.

10. Diaper, D., Knowledge Elicitation: Principles, Techniques, and Applications, Ellis Horwood, Chichester, U.K., 1989, pp. 25-28.

11. Parsaye, K., and Chignell, M., Expert Systems for Experts, John Wiley & Sons, New York, NY, 1988, pp. 291-292.

12. Slagle, J., Gardiner, D., and Han, K., "Knowledge Specification of an Expert system.", IEEE Expert, August 1990, pp. 29-38.

13. Wilson, M., "Task Models for Knowledge Elicitation." in Knowledge Elicitation: Principles, Techniques and applications, ed. D. Diaper, Ellis Horwood Ltd. Chichester, U.K, 1989, pp. 197-219.

14. Johnson, L., and Johnson, N., "Knowledge Elicitation Involving Teachback Interviewing.", in Knowledge Acquisition for Expert Systems: A practical Handbook, ed. A. Kidd, Plenum Press, London, U.K., 1987, pp. 91-108.

15. Clancey, W.J., "Heuristic Classification", Artificial Intelligence, Vol 27, 1985, pp. 209-350.

16. McDermott, J., "Preliminary Steps Towards a taxonomy of Problem-Solving Methods." in Automating Knowledge Acquisition for Expert Systems, ed. Marcus, S., Kluwer Academic Publishers, Boston, 1988, pp.225-255.

17. Olson, J., and Rueter, H., "Extracting Expertise from Experts: Methods for Knowledge Acquisition.", Expert Systems, Vol 4, No 3, Aug 1987, pp. 152-168.

18. Hart, A., "Knowledge elicitation: issues and methods.", Computer-Aided Design, Vol 17, No 9, Nov 1985, pp. 455-462.

19. Hoffman, R.R, "The problem of extracting the knowledge of experts from the perspective of experimental psychology", AI Magazine, Summer 1987, pp. 53-67.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

A NEURAL NETWORK APPLICATION TO SEARCH AND RESCUE SATELLITE AIDED TRACKING (SARSAT)

Ivan W. Taylor[1] and Michel O. Vigneault[2]

Abstract

For the past four years, the Operational Research Advisors at Air Transport Group Headquarters have been working on heuristic methods to predict the location accuracy category of SARSAT hits based on the parameters of the Doppler curve fit. The current heuristics that have been developed and implemented work reasonably well but are far from ideal. They consistently have difficulty identifying very good solutions when they occur and very poor solutions when they occur. Normal data fitting techniques based on multiple linear regression do not work well because the SARSAT data is inherently non-linear. Therefore, neural networks were applied to the SARSAT data as a non-linear data fitting technique. Although the results so far have been promising, they are still far from ideal. Other artificial intelligence technologies, such as fuzzy logic, inductive learning and expert systems are being investigated to enhance the neural network approach.

[1]Operational Research Advisor, [2]Operational Research Advisor 2, Air Transport Group, Air Transport Group Headquarters, Astra, Ontario

2

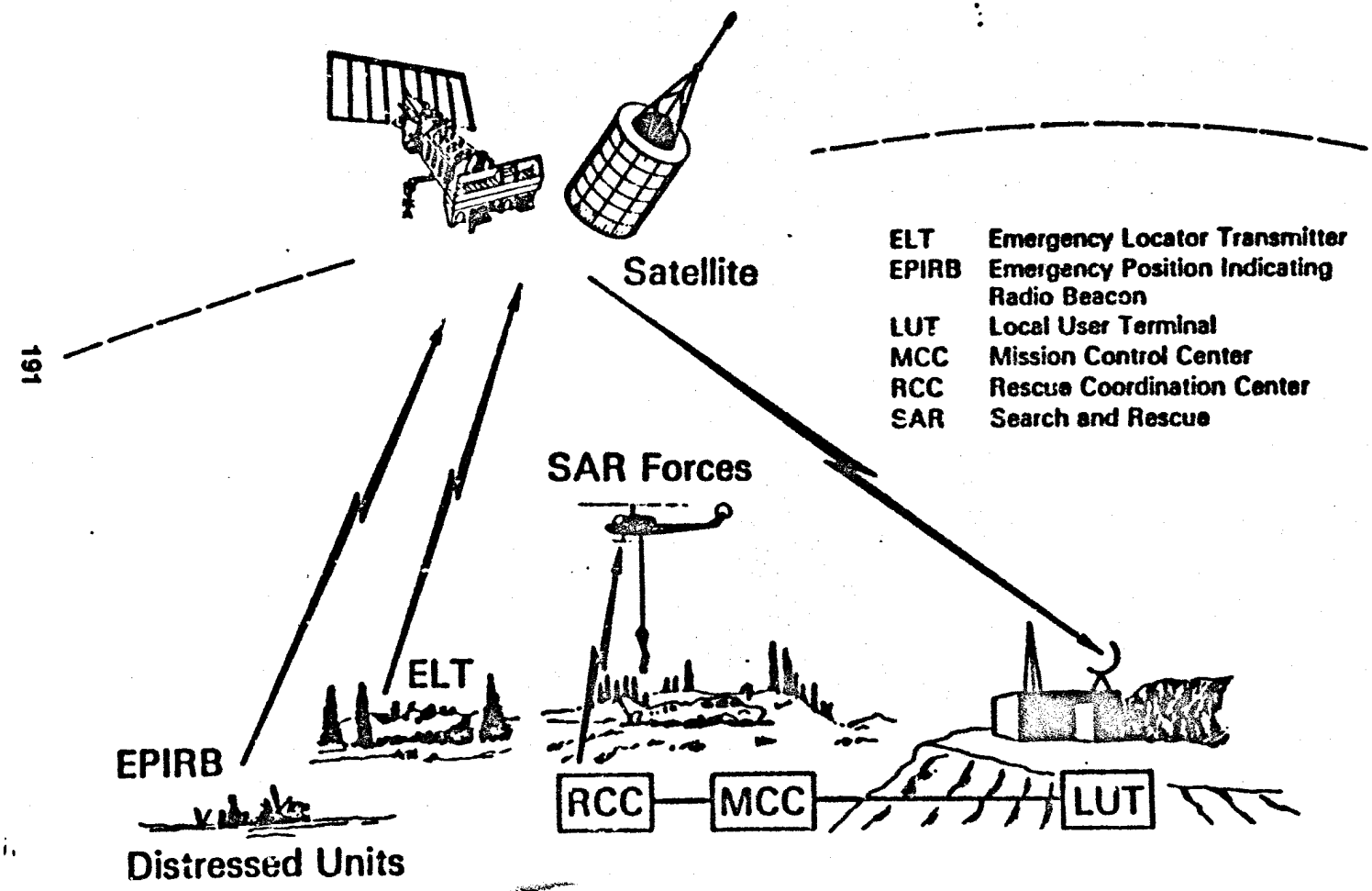
INTRODUCTION

The Search and Rescue Satellite Aided Tracking (SARSAT) system uses polar-orbiting satellites to monitor the surface of the earth for Emergency Locator Transmitters (ELTs). If one is detected, its signal is retransmitted to ground facilities, called Local User Terminals (LUTs), for processing. The location of the ELT is determined by fitting a Doppler curve to the signal and this location estimate is sent to the Mission Control Center (MCC) for retransmission to a foreign MCC or a national Rescue Coordination Center (RCC) for action (see Figure 1).

When the SARSAT system was first introduced in Canada in 1985, it suffered a number of growing pains. One particular problem was false alerts on the 121.5 MHz frequency which were caused by spurious voice communication on the channel even though it is supposed to be reserved for emergency broadcasts. The Operational Research Branch at Air Transport Group worked on a filtering procedure to screen out these false alerts and prioritize the ELT signals. This resulted in the 'confidence factor' method. The confidence factor was also useful as a predictor of the accuracy of the location estimate. Studies in 1987 and 1988 [1, 2 and 3] developed and refined a heuristic based on the Doppler curve fit parameters. This heuristic was later implemented in the LUT software.

Unfortunately, the current heuristic is not well understood. It is based on empirical data rather than a sound theoretical foundation and the results are far from ideal. Any new method should have better performance, be easier to understand and have a more rigorous theoretical foundation. A new method will be considered in this report based on the artificial intelligence technique called neural networks [4]. We have obtained a program developed by Dr. Simon Barton from Defence Research Establishment Suffield which simulates a small neural network [5]. We will 'train' the neural network on a sample of 173 observations. Then we will 'test' the method on a sample of 172 different observations and compare the results to the current method.

FIGURE 1: Basic Concept



4

MODEL COMPARISON METHOD

The confidence factor (CF) method that has been implemented is tied to the International Telecommunications Union (ITU) standards for reporting position error:

a. CF=4, 'A' category, within 5 nm;

b. CF=3, 'B' category, within 20 nm;

c. CF=2, 'C' category, within 50 nm; and

d. CF=1, 'D' category, greater than 50 nm in error.

We will combine the 'C' and 'D' categories to increase our sample size for this subset and define it as 'errors greater than 20 nm'. The confidence factor methods will attempt to predict the location error category.

In [6], we developed a statistical technique for evaluating the accuracy of a confidence factor method for a set of observations. We first divided the data set into four overlapping subsets based on the actual location error: very good solutions (less than 5 nm in error), good solutions (less than 20 nm in error), poor solutions (greater than 5 nm in error) and very poor solutions (greater than 20 nm in error). Notice very good solutions are also good, very poor solutions are also poor, and some good solutions can be considered poor depending on your point of view (see Figure 2).

We can display the accuracy of the confidence factor method by plotting the percentage of the time good (and very good) solutions are identified when they occur against the percentage of the time very poor (and poor) solutions are identified when they occur. If the confidence factor method calls a good solution very poor, or a very good solution poor, we call that a Type I mistake (or underconfidence). If the confidence factor method calls a poor solution very good, or a very poor solution good, we call that a Type II mistake (or overconfidence). This dichotomy is similar to producer's and consumers' risk in industrial testing [7]. When these values are plotted, we can easily determine an ideal confidence factor model from an underconfident, overconfident or just plain poor model (see Figure 3).

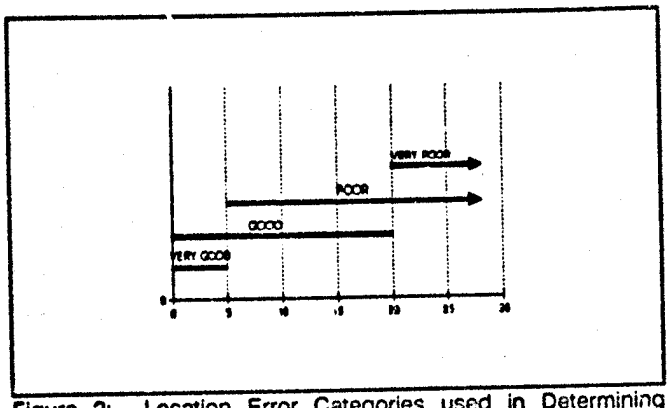


Figure 2: Location Error Categories used in Determining Confidence Factor Accuracy in Nautical Miles

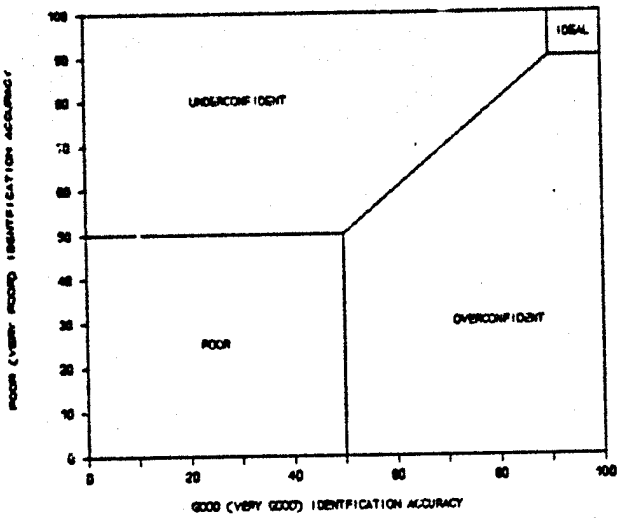


Figure 3: A Method of Evaluating Confidence Factor Models

THE DATA

For each observation, we have nine independent variables and one dependent variable. The following description of the independent variables is taken from [2]:

a. The satellite (SAT). There are four different satellites.

b. Major and Minor Axes (MAJ and MIN). These are the major and minor axes of the estimated error ellipse. They are used together with the angle of inclination of the major axis with respect to true north to give an area in which the ELT is presumably located a certain percent of the time.

c. The probability (PROB). This is the probability that the solution identified is the correct one and not its image with respect to the satellite track.

d. Bias. Bias is the difference in signal frequency of the beacon from the 121.5 MHz standard frequency. It is useful when wanting to combine several detections from the same area into one case. If the bias from two detections are within .600 KHz, and from the same area, they will be merged into one case.

e. Cross Track Angle (CTA). CTA is a measure of the angle between the satellite's orbital plane and a vector from the centre of the earth to the ELT at the time the spacecraft is closest to the beacon. A CTA close to 0° occurs when the satellite passes almost directly overhead, whereas CTAs greater than 16° are near the satellite's horizon.

f. Standard Deviation (SDEV). SDEV is a measure of how well the frequency data composing a Doppler curve fits the model equation defining it. It could probably be more accurately defined as a goodness of fit measure, with the smaller the SDEV, the better the fit.

g. Number of Points (NPTS). NPTS is directly related to how long the signal was tracked. For 121.5 MHz signals, one point is generated every two seconds the signal is tracked. Once the LUT loses the signal, all of the points are used to make up one detection.

h. Normalized Quality Factor (NQF). NQF is the ratio of Quality Factor (QF) to NPTS, where QF is a function of the strength of the signal and the length of time that the signal was received.

The dependent variable is the location error category.

PERFORMANCE OF THE CURRENT METHOD

Our data set consists of 345 observations: 150 (43%) less than 5 nm in error; 129 (37%) between 5 and 20 nm in error; 66 (19%) greater than 20 nm in error. We will divide this into two separate sets: a training set consisting of the first 173 observations and a test set consisting of the next 172 observations. Table I shows the classification of test data using the current method.

TABLE I

CLASSIFICATION OF TEST SET BY CURRENT CONFIDENCE FACTOR METHODOLOGY

		ACTUAL CATEGORY			TOTAL
		A <5nm	B 5< <20nm	C, D >20nm	
PREDICTED CATEGORY	CF=4	26	12	2	40
	CF=3	37	43	16	96
	CF=2,1	4	13	19	36
	TOTAL	67	68	37	172

As expected, the current method does much better than a random assignment based on the relative distribution of the solutions by location error category. Table II shows the percentages that would be expected from a random assignment. Table III shows the percentage obtained by the current confidence factor method. The higher values on the diagonal in Table III compared to Table II show that the current method is working.

Table IV shows the accuracy of the current method in terms of Type I accuracy and Type II accuracy. Table V provides 95% confidence intervals on the accuracy using the technique described in [8]. Figure 4 shows the accuracy of these results graphically. We can see that the current method has difficulty distinguishing very good solutions when they occur (39% accuracy) and also has difficulty identifying very poor solutions when they occur (51% accuracy). For the 5 nm breakpoint, the current method is underconfident and for the 20 nm breakpoint the current method is overconfident. This is an undesirable situation.

TABLE II

EXPECTED PERCENTAGES THAT WOULD BE OBTAINED BY RANDOM ASSIGNMENT

		ACTUAL CATEGORY		
		<5nm	5< <20nm	>20nm
PREDICTED BY	<5	15.2	15.4	8.4
RANDOM	5< <20	15.4	15.6	8.5
ASSIGNMENT	>20	8.4	8.6	4.6

TABLE III

PERCENTAGES OBTAINED BY CURRENT CONFIDENCE FACTOR METHOD

		ACTUAL CATEGORY		
		<5nm	5< <20nm	>20nm
PREDICTED	<5	15.1	7.0	1.2
CATEGORY	5< <20	21.5	25.0	9.3
	>20	2.3	7.6	11.0

TABLE IV

ACCURACY OF CURRENT CONFIDENCE FACTOR METHOD

SOLUTIONS <5NM IN ERROR			SOLUTIONS >5NM IN ERROR		
TOTAL	CORRECT CF=4	UNDER-CONFIDENT CF=3,2,1	TOTAL	OVER-CONFIDENT CF=4	CORRECT CF=3,2,1
67	26 (38.8%)	41	105	14	91 (86.7%)
SOLUTIONS <20NM IN ERROR			SOLUTIONS >20NM IN ERROR		
TOTAL	CORRECT CF=4,3	UNDER-CONFIDENT CF=2,1	TOTAL	OVER-CONFIDENT CF=4,3	CORRECT CF=2,1
135	118 (87.4%)	17	37	18	19 (51.4%)

TABLE V			
CONFIDENCE INTERVALS ON CURRENT METHOD ACCURACY			
SOLUTION CATEGORY	LOWER BOUND	ESTIMATE	UPPER BOUND
<5nm in Error	27.14	38.81	51.50
>5nm in Error	78.64	86.67	92.51
<20nm in Error	80.61	87.41	92.49
>20nm in Error	34.40	51.35	68.08

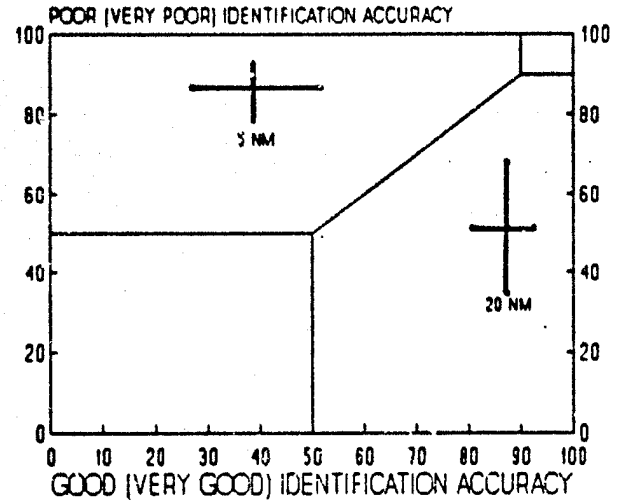


Figure 4: Performance of Current Method on Test Data

It is apparent from these results that the current method leaves much room for improvement. We will now develop a neural network for predicting location error category based on the training set. Then we will examine the ability of the neural network model to generalize to new cases by applying it to the test set and compare the neural network results to the results of the current method.

PREDICTING LOCATION ERROR CATEGORY

One problem should be noted. The neural network method produce numbers with a fractional component. The actual category, on the other hand, is an integer. Therefore we must convert these predictions to integers. After considerable thought, we decided to simply round the predicted values. Therefore a predicted value of 3.9 would be rounded to 4 and a predicted value of 3.4 would be rounded to 3. However, since we have grouped category C and D together, we will call all predicted values less than 2.5, 2. Similarly, any prediction greater than 3.5 will be called 4.

A neural network model was developed using the 173 observations training set. When this model was used to classify the observations in the test set, the results were as shown in Table VI. Tables VII and VIII provide more details on the accuracy of the neural network method. Figure 5 shows the accuracy of these results graphically.

TABLE VI

NEURAL NETWORK RESULTS FOR TEST SET

		ACTUAL CATEGORY			TOTAL
		A <5	B 5<<20	C,D >20	
PREDICTED CATEGORY	CF=4	44 (25.6)	15 (8.7)	7 (4.1)	66
	CF=3	21 (12.2)	43 (25.0)	14 (8.1)	78
	CF=1,2	2 (1.2)	10 (5.8)	16 (9.3)	28
TOTAL		67	68	37	172

TABLE VII					
ACCURACY OF NEURAL NETWORK METHOD					
SOLUTIONS <5NM IN ERROR			SOLUTIONS >5NM IN ERROR		
TOTAL	CORRECT CF=4	UNDER-CONFIDENT CF=3,2,1	TOTAL	OVER-CONFIDENT CF=4	CORRECT CF=3,2,1
67	44 (65.7%)	23	105	22	83 (79.0%)
SOLUTIONS <20NM IN ERROR			SOLUTIONS >20NM IN ERROR		
TOTAL	CORRECT CF=4,3	UNDER-CONFIDENT CF=2,1	TOTAL	OVER-CONFIDENT CF=4,3	CORRECT CF=2,1
135	123 (91.1%)	12	37	21	16 (43.2%)

TABLE VIII			
CONFIDENCE INTERVALS ON NEURAL NETWORK ACCURACY			
SOLUTION CATEGORY	LOWER BOUND	ESTIMATE	UPPER BOUND
<5nm in Error	53.06	65.67	76.85
>5nm in Error	70.01	79.05	86.38
<20nm in Error	84.99	91.11	95.32
>20nm in Error	27.10	43.24	60.51

As we can see from Figure 5, the neural network model performed quite well on the test set. In fact, the neural network results are significantly better at identifying very good solutions when they occur (less underconfidence). That is, when we compare the "<5 nm" results for the current method (Table V) and for the neural network method (Table VIII), we see that the confidence intervals do not overlap which indicates a statistically significant improvement. None of the other differences are statistically significant.

The neural network model based on location error category shows promise. The results appear to demonstrate a significant improvement in Type I accuracy for very good solutions (i.e. reduction in underconfidence) without a reduction in Type II accuracy (i.e. without introducing more

overconfidence). Although the results are still far from the ideal, this area of research should be pursued with more sophisticated tools and larger data sets.

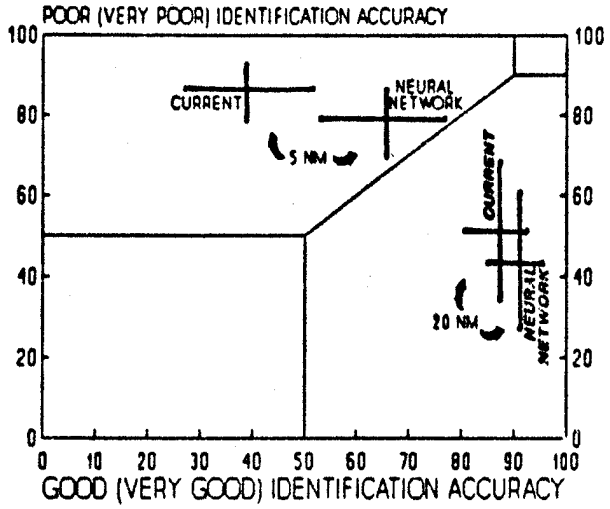


Figure 5: Performance of Neural Network Method on Test Data

CONCLUDING REMARKS

We found that the current confidence factor method, although performing better than random assignment, has the undesirable characteristic that it has difficulty identifying very good solutions or very poor solutions when they occur.

We examined a new methodology based on neural networks. The neural network method showed significantly improved ability to identify very good solutions when they occurred compared to the current method. It did not, however, produce improvements in terms of identifying very poor solutions when they occur. Although the neural network results show promise, it is still far from the ideal.

A consultant company, ABIT Systems Inc., has proposed a research contract to apply interval mathematics (fuzzy set theory) and inductive learning as preprocessing before applying neural networks to this problem [9]. These more sophisticated techniques, which are not currently available to the Operational Research Advisor, may enhance the modest improvements that have been demonstrated in our research and may take us another step towards an ideal system.

REFERENCES

1. Hunter, Charles J., "A Critical Review of COSPAS/SARSAT Data Categorization and Location Accuracy Models", ATGOR Staff Note 1/87, August 1987.

2. Riehl, Christopher M., "COSPAS/SARSAT Location Accuracy Models for 121.5 MHz Distress Signals" ATGOR Staff Note 3/87, August 1987.

3. Hunter, Charles J., "COSPAS/SARSAT Canadian Location Accuracy Models for 121.5 MHz Distress Beacons: Revisited", ATGOR Staff Note 1/88, April 1988.

4. Wasserman, Philip D., "Neural Computing: Theory and Practice", Van Nostrand Reinhold, New York, 1989.

5. Barton, Simon A., "A Technique for Optimizing a Neural Network", Suffield Memorandum No. 1351, Defence Research Establishment Suffield, December 1990.

6. Taylor, Ivan and Vigneault, Michel, "Preliminary Confidence Factor Analysis" ATGOR Internal Working Paper 2/91, May 1991.

7. Johnson, N.L. and Leone, F.C., "Statistics and Experimental Design in Engineering and the Physical Sciences", Volume I, Wiley and Sons, New York, 1964.

8. Taylor, Ivan, "Confidence Interval Techniques for the 1990 COSPAS/SARSAT Exercise Analysis", :ATGOR Staff Note 3/91, June 1991.

9. Turksen, I. and Hong, Z., "Artificial Intelligence Applications to the Canadian SARSAT Location Accuracy Model", Contract Proposal, July 1991.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

INSPECTION OF DND WARREN TRUSS BUILDINGS USING KBES IN A WINDOWING ENVIRONMENT

H. C. Fu[1] and G. Akhras[2]

Abstract

The Department of National Defence (DND) has over two hundred timber Warren truss hangars throughout the country. The roof of these structures is supported by timber Warren trusses. These structures were built in the early 1940's and are showing signs of deterioration. In order to properly maintain these aging structures, regular inspections by military engineers and technologists to assess their structural behaviour are required.

Inspection of a building often requires following guidelines and applying heuristic knowledge such as experience and rules of thumb. Because of the nature of the inspection process and all the heuristic knowledge that is associated with it, Warren truss inspection lends itself well to an expert system application.

The development of a knowledge based expert system for the maintenance of timber Warren trusses is now underway. The system combines expert system technologies, object-oriented programming, relational database models and hypertext/graphics in a windowing environment. This paper presents a brief insight into the strategy and technique used in the development of the system. Two sample consultations are given to demonstrate the capabilities of the system.

[1]Research Engineer, [2]Associate Professor, Dept of Civil Engineering, RMC, Kingston, Ontario

2

INTRODUCTION

An expert system, also known as knowledge-based expert system (KBES), can be considered as a computer program that captures human knowledge and decision making process. An expert system has two basic components: a knowledge base and an inference engine. The knowledge obtained from the human expert or experts comprises information specific to the domain of the problem being addressed and is captured in the knowledge base. The inference engine interprets and applies the knowledge base and attempts to make decisions to problems that would ordinarily require a human expert. Fully developed expert systems are capable of accepting facts from the user, processing these facts against the knowledge base, and, on the basis of these facts and knowledge, making conclusion which are close to the conclusion of a human expert.

Applications of expert system technology can be dated back to the mid-1970's. MYCIN [9], widely regarded as the first successful expert system, was developed at Stanford University to help diagnose and treat glaucoma-related infections. Since then, expert systems have been used in a wide variety of domains ranging from medical diagnosis to military strategic assessment, and from nuclear reactor control to information management. The potential uses of expert systems appear to be virtually limitless. The recent interest and potential prospects in the development and application of the expert system technology in the fields of engineering are both enormous [1,3,6] and growing rapidly.

A system for the overall management of timber Warren truss structures is being developed by the KBES Group of the Department of Civil Engineering at the Royal Military College of Canada [2]. This system has many sub-systems related to the maintenance of all DND timber Warren truss buildings. One of which is the diagnosis or inspection of the trusses. The principle objective of the inspection process is to determine the appropriate action required to repair defective members of the structure. Some of the solution strategies and development techniques of the system are presented in this paper. Two sample sessions are included to demonstrate the capabilities of the system.

3

TIMBER WARREN TRUSSES

The Structure

DND owns many hangar type timber Warren truss structures on CF bases throughout the country. These Warren truss buildings, built during and soon after World War II to meet the needs of the expanded Armed Forces establishment, were constructed as temporary structures and were used by the military mainly for aircraft housing and maintenance. It is estimated that across Canada, there are more than two hundred of these 'temporary' buildings still standing.

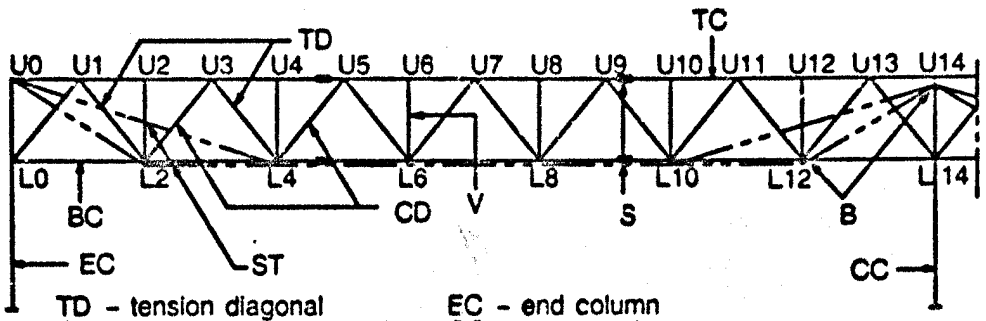
The buildings were designed and built in two truss configurations, namely an eight-panel pitched chord truss and a seven-panel parallel chord truss. The overall configurations of the two types of trusses were standardized with a span of 34.16 m. A standard building has eleven such trusses spaced at 4.88m for a total length of 48.8 m. Fig. 1 shows a double Warren truss with the seven-panel parallel chord configuration.

There are three types of buildings branching from the two truss types: single pitched, single parallel and double parallel buildings in which two single parallel chord trusses were joined together side by side sharing a central column. Plan view of a typical double parallel building is illustrated in Fig. 1b. Only a few of the hangars still standing today used the pitched Warren truss and they all are single span buildings.

The Problem

Due to the large number of buildings involved and the urgent demand for timber, improperly cured timber was used in some structures. Within the first few years of service and after the timber dried out, the trusses began to show signs of distress due to shrinkage. The resulting structural deficiencies included cracking and splitting of truss or splice members, fracture of truss members, excessive deformation of trusses and loss of camber.

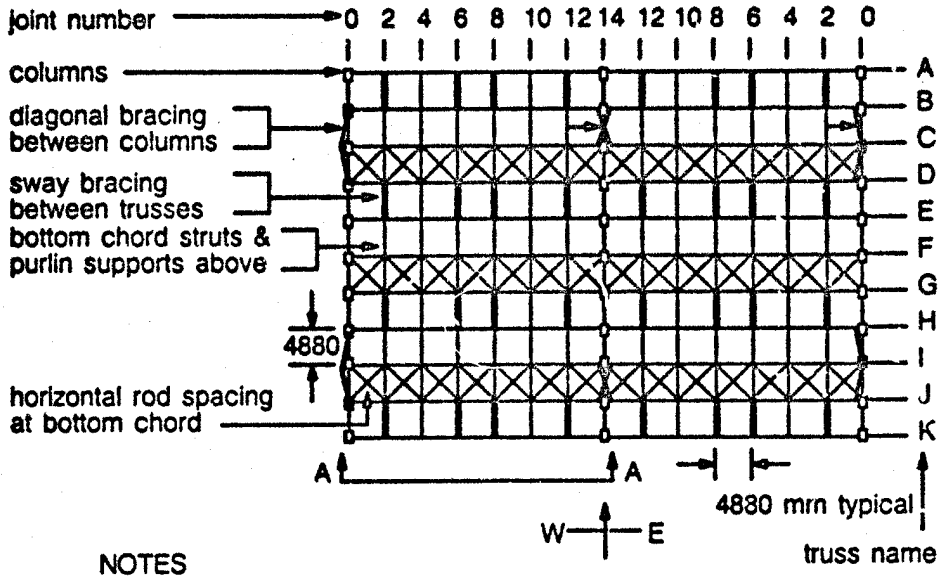
Shortly after World War II, the structural deficiencies were recognized and a program of rehabilitation, repair and reinforcement was established. Structural analysis of



TD - tension diagonal
CD - compression diagonal
V - vertical
TC - top chord
BC - bottom chord

EC - end column
CC - centre column
B - turning block for strands
S - splices
ST - four 13 mm strands (new system)
- eight 8 mm strands (old system)

(a) section A-A Elevation of a seven panel parallel Warren truss



NOTES

1. Location and configuration of tie rod bracing, sway bracing and diagonal bracing varies from building to building.
2. The number of trusses varies from building to building.

(b) Plan view of a double parallel Warren truss building

Fig.1 A Double Width Seven-Panel Parallel Chord Building

the truss indicated that, for buildings where the live load, mainly the roof snow load, was high, the truss end panels and the bottom chord were overstressed. A system of strengthening by post-tensioning the Warren trusses was introduced to relieved some of the stresses in these areas (Fig. 1a).

To avoid inconsistent results and to eliminate considerable duplication of effort, DND established guidelines and direction for the assessment, repair and maintenance processes of Warren truss buildings. It was this policy that led to the publication of the Construction Engineering Technical Order, or CETO [4]. CETO was first published in 1967 and later updated in 1988.

CETO can be considered as a three-part maintenance manual. The first part provides instructions for the inspection of Warren truss buildings. The second part provides guidelines for determining the necessary reinforcement work in upgrading an existing Warren truss building to meet the requirements of the National Building Code of Canada [8]. The last part of CETO provides specifications for the repair and reinforcement work required because of the current or existing condition of the building.

The process of inspecting a building as outlined in CETO often requires following guidelines and applying heuristic knowledge such as experience and rules of thumb. Over the years, DND has developed the necessary knowledge and structural techniques adapted specifically to timber Warren trusses. This accumulating expertise is dispersed among very few engineers and technologists. Extensive consultation is often required to reach an appropriate decision. In addition, DND does not have sufficient personnel with necessary knowledge to carry out, as often as needed, the task of inspecting deteriorating buildings.

KBES IN WINDOWS ENVIRONMENT

An expert-systems development software called Level5 Object [5] is used to develop a knowledge based expert system for the inspection of timber Warren trusses. The system combines expert system technologies, object-oriented programming, relational database

models and hypertext/graphics capabilities in a flexible windowing environment.

Object-Oriented Programming

Level5 Object is an object-oriented expert system development shell, combining the versatility of object-oriented techniques with multiple inferencing strategies in windowing environment. It provides an interactive windows-based user interface integrated with Production Rule Language (PRL), one of the many knowledge representation schemes used in expert systems.

In object-oriented environment, the classification and organization of data within a software application are modeled in the same way as human normally classify and organize knowledge. Data and programs are represented as objects. In other words, an object is the sum of its data and procedures. Instead of passing data as parameters as in conventional programming environment, the objects perform operations on themselves.

Objects are definitions of database-like entities with associated relationship references, procedures, and actions. An object's structure is defined by its class and attribute declarations. An instance is a specific occurrence of a class and holds data values of its attributes. Fig.2 depicts the structural breakdown of a Level5 Object object.

To understand and process a vast amount of information, objects are grouped into a class which defines the properties, inheritance, and attributes of an object. Properties of a class define its inheritance, its source and whether it is limited to a single or multiple instances.

A class describes the structure and behaviour of an object within an application and is defined by a collection of characteristics called attributes. Level5 Object attributes consist of a name, a type, facets, methods, rule group and demon group. Attribute type specifies the type of information associated with the class such as numeric data, a true/false statement or graphics data that constitutes a picture. Facets control how the inference engines process and use attribute. Methods establish developer-defined procedures

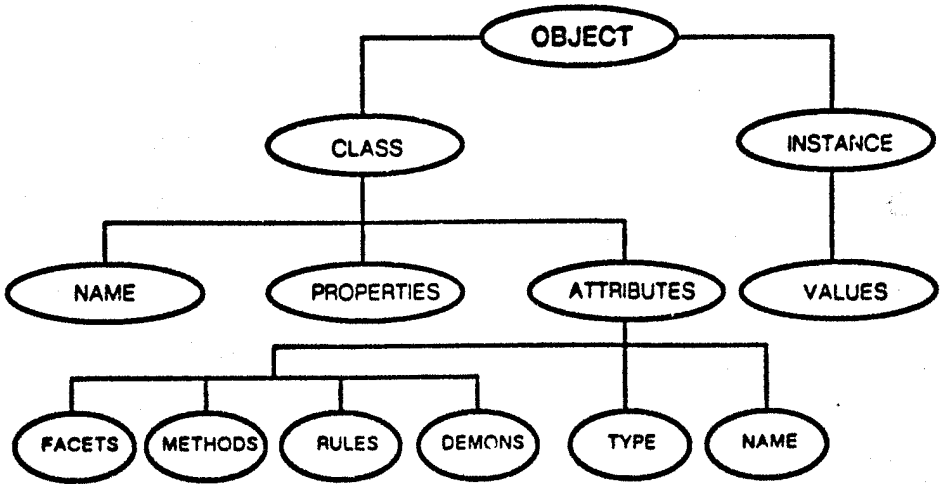


Fig. 2 Components of a Level5 Object object

associated with each attribute. Rule group uses backward-chaining rules to conclude the same attribute. Demon group uses forward-chaining demons to reference the same attribute.

Objected-Oriented Database Management

Level5 Object supports direct access to external programs and databases with a management system that integrates and controls the interaction between the knowledge base and the database files. Direct database access enables Level5 Object to read and write to files directly from within the knowledge base. Level5 Object views these databases as objects, which are referenced and manipulated with standard PRL grammar.

Windows Environment

Level5 Object provides an interactive, windows-based graphical user interface. Applications developed with Level5 Object run under and follow the conventions of Microsoft Windows [7]. The user interface is invoked through Microsoft Windows and runs in a fully triggered environment. Pointing and clicking an icon or rule name automatically activate the appropriate editor or browser. Clicking on a hyperregion gives end users

auxiliary text and graphic information as needed.

Using the clipboard of Windows to transfer files, paints and graphics can be incorporated into a Level5 Object display. Level5 Object can also display photo or graphic images from image files created by other image processing software as long as these files are compatible with Microsoft Windows.

Level5 Object also supports the hypermedia paradigm. By clicking on a hyperregion in a window or dialogue box, another window can be opened containing additional information in text and graphic formats.

KBES FOR THE INSPECTION OF DND WARREN TRUSS BUILDINGS

The Objective

To accommodate the nature of maintenance of a timber Warren truss, a knowledge-based expert system for the diagnosis of the trusses is being developed. This system groups many sub-systems related to the diagnosis, repair and database management of the rehabilitation history of these buildings. The scope of this project belongs to the class of integrated systems which includes many independent but related sub-systems, and which needs continuous development, updating and revision.

Expert knowledge on Warren truss inspection is held by a few, and among them, no one has an in-depth knowledge of the whole field. The system is intended to be used as a guide for the user, either experienced or novice, throughout the inspection process. Implementation of the system reduces the dependence on specialist staff for routine inspection work and serves to transfer the diagnosis knowledge from the experts to local maintenance personnel as well as among the experts themselves.

The System

The system is being developed to run on personal computers using the expert system shell Level5 Object which operates under Microsoft Windows environment. The

system can be compiled and encrypted to create a run-only system. The run-only system can then be delivered to the end users as the completed application. This run-only system is smaller, more efficient and portable.

The knowledge is drawn mainly from the maintenance manual, i.e. CETO, and from several engineers who were previously or are presently involved with the inspection and the evaluation of the structural behaviour of these structures. Using Level5 Object, the domain experts' knowledge are translated into rules and facts while following a special syntax. The same group of experts from whom heuristic knowledge is drawn, will also be involved in the process of testing and validating the system.

Facts required by the system is the user's visual observations input by answering questions related to each element. The system interfaces to the user and has an efficient explanatory component to make the comprehension and checking of how a solution is reached possible and effective. After putting facts into the system, the user receives diagnoses and recommendations and the user is responsible for making use of it.

The output lists the recommendations for action. They includes identifying all members requiring replacement or repair along with the reasoning for the remedial action. Because the main source of knowledge of this system is the maintenance manual, specific references to particular explanations and details in this manual, if needed, are provided to assist the user.

EXAMPLES

If an expert system is to be accepted, it should exhibit interactive graphics and simulation facilities that increase the end user's understanding and control of the system being represented. The system being developed is highly user friendly with many graphics-oriented interface features such as interactive graphics, window management, explanation expansion and graphical representation of knowledge base by mapping graphic displays to and from conclusions.

Since the system has graphical user interface with explanatory facilities, little or no programming knowledge and experience is required to conduct an inspection of timber Warren trusses. Users simply point and click his/her way through the diagnosis process. Two sample consultations with the system regarding the inspection of truss vertical members and purlins, are presented in the following sections.

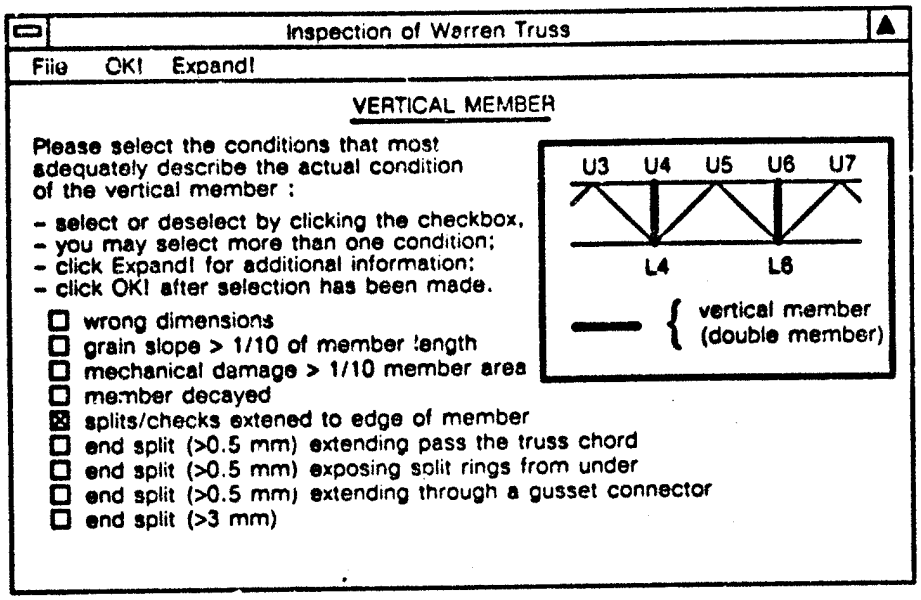
Truss Vertical Member

Fig. 3 illustrates the typical screens ((a) to (d)) during a consultation session with the system. The member under consideration is the vertical member. Graphical display of part of the truss with locations and identifications of vertical members are displayed to assist the user in responding to the question as shown in screen (a).

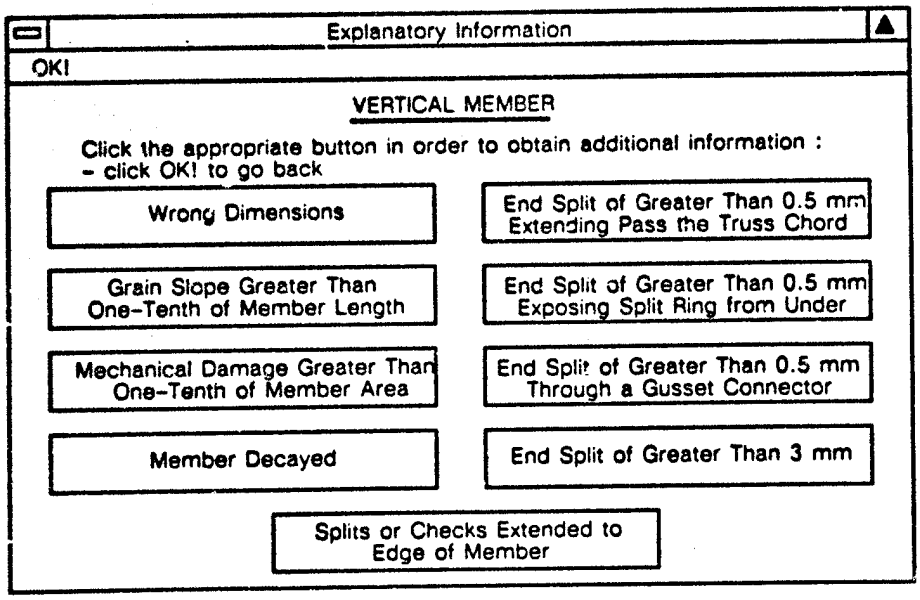
If the user decides to want to know more.about one or more particular items among the given checklist, he/she can simply click the Expand! key on the menu bar shown in screen (a). Clicking the Expand! key directs the user to the 'help' window as shown in screen (b). The user can then point to and click the appropriate object button in screen (b) to activate the help facility. Screen (c) shows additional information as requested by the user with graphical and photo image displays to enhance the presentation of the knowledge.

Many of the graphic displays in the system illustrate more than one condition of defects. For example, screen (c) illustrates three different types of defects, namely, check/split extending to edge of member, split passing through a connector and split in the splice block extending through a connector. Screen (c) also demonstrates how to measure the slope of grain. Whenever one of these four objects were activated by clicking the object button in another 'help' window similar to screen (b), screen (c) will appear.

Screen (d) is the conclusion display. In this case, the user has indicated that the member has "checks/splits extending to the edge of the member". Recommendation from the system suggests that the member be removed and replaced with details of the remedial action being shown on the same display.

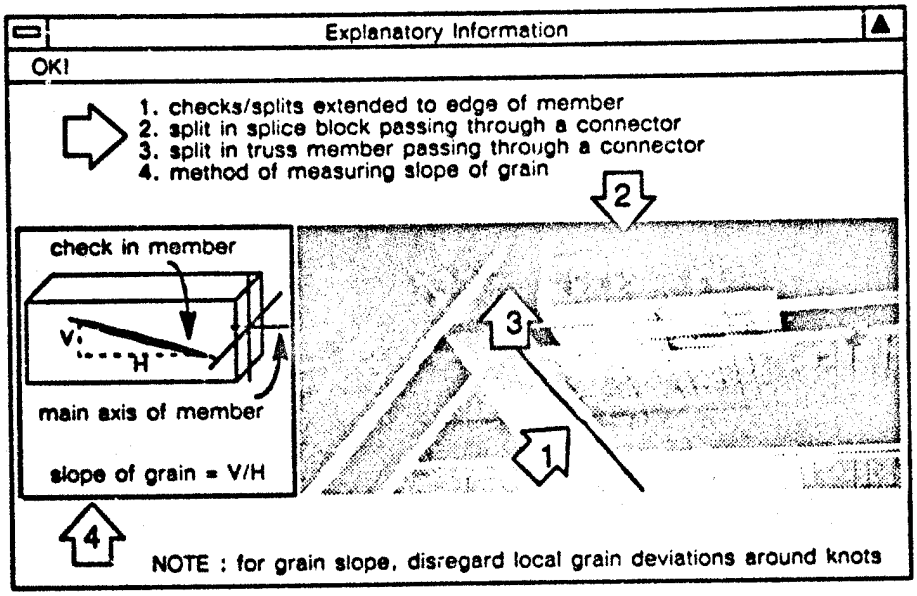


(a)

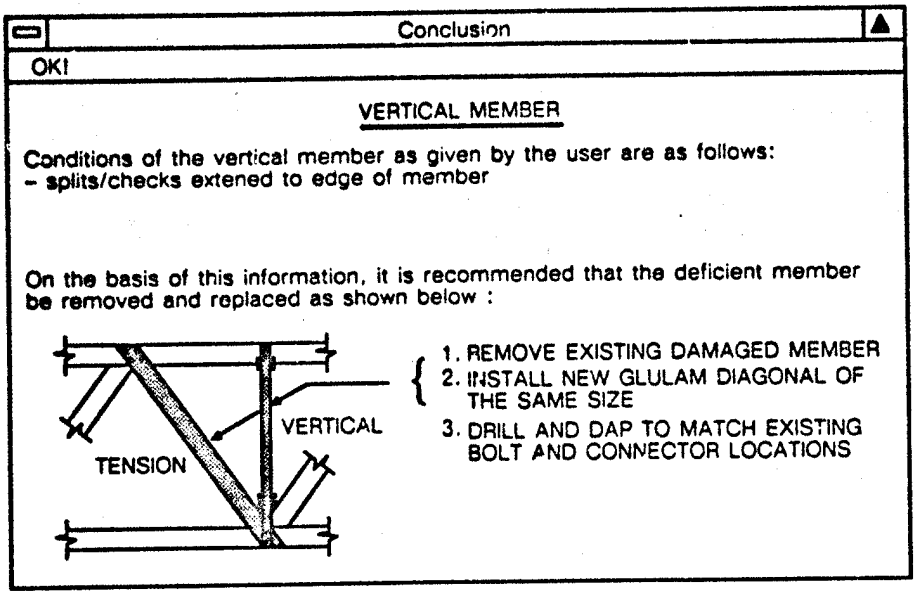


(b)

Fig.3 Screen Displays for Vertical Member



(c)



(d)

Fig.3 Screen Dispiays for Vertical Member (cont.)

Purlin

Fig. 4 is representative of typical screens ((a) to (d)) for the diagnosis of a roof purlin. To assist the user in deciding what is meant by flexural failure, a photo of an actual case, screen (c), is displayed right in front of the user after the user has clicked the Expand! key in screen (a) and the appropriate button in screen (b).

Screen (d) is the conclusion display suggesting that this member be replaced. Details of the reinforcement for damaged purlins are also shown on the same display.

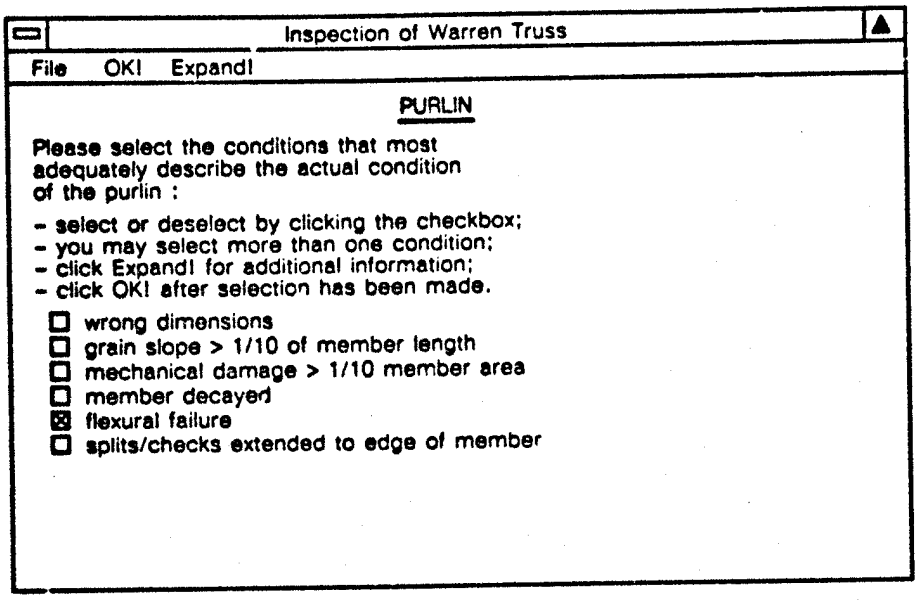
Remarks

At the end of each session, the user can save the context of the session by appending records to a database using the values currently in the context. For example, records on truss vertical member can be appended to an existing database file which contains the history of problems and remedial actions on all vertical members. Since the system can directly communicate with existing conventional software such as databases and computational/analytical programs, the user can examine the history of any particular member either at the start or at the end of each session.

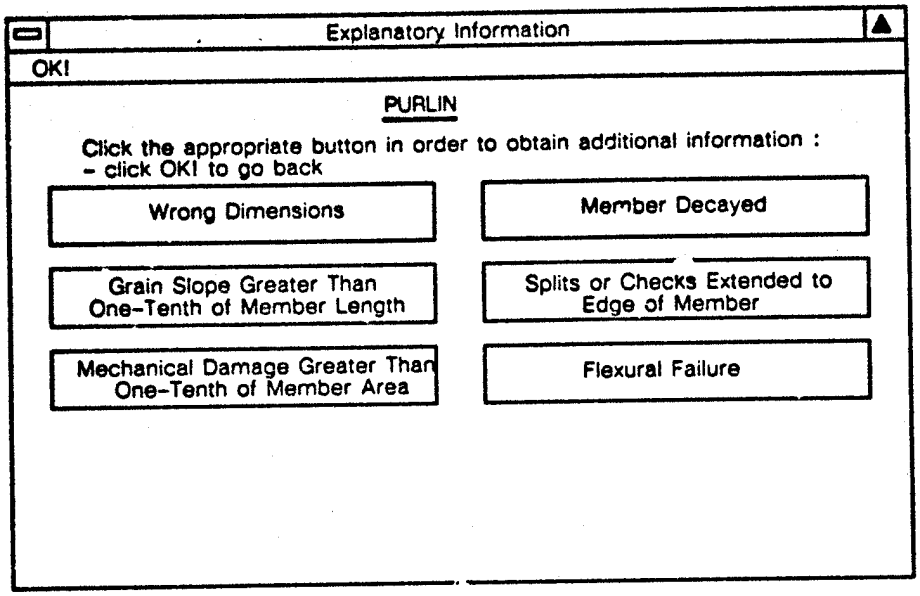
CONCLUSIONS

Inspection of timber Warren trusses requires following guidelines and applying heuristic knowledge such as experience and rules of thumb. Because of the nature of the inspection process and all the heuristic knowledge that is associated with it, Warren truss inspection was found to be a good application of expert system technology.

The proposed system combines expert system technologies, object-oriented programming, relational database models and hypertext/graphics in a windowing environment. Strategy and technique used in the development of the system were presented and discussed. Two sample consultations are given to demonstrate the capabilities of the system.

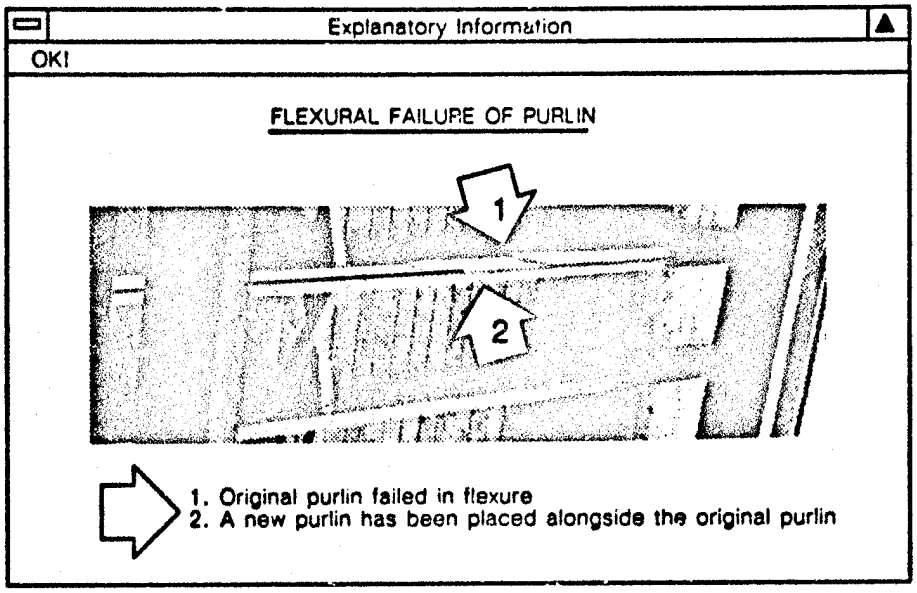


(a)

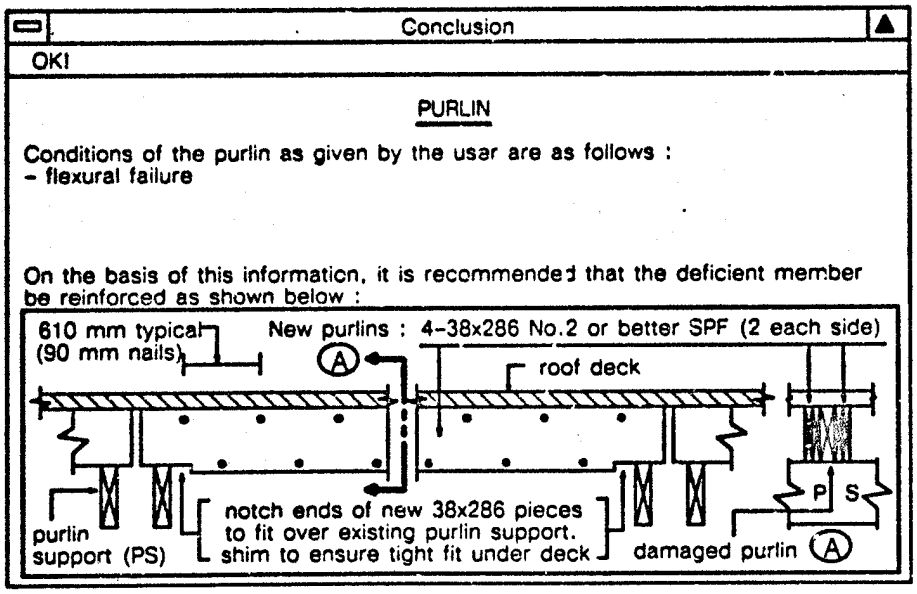


(b)

Fig.4 Screen Displays for Purlin



(c)



(d)

Fig.4 Screen Displays for Purlin (cont.)

ACKNOWLEDGEMENTS

The authors would like to thank Computing Services of the Royal Military College of Canada. The financial assistance of the Department of National Defence of Canada is gratefully acknowledged.

REFERENCES

1. Adeli, H., "Expert Systems in Construction and Structural Engineering", Chapman and Hall, New York, 1988, 330 p.
2. Akhras, G., "Computer Diagnosis of Timber Warren Trusses", Proceedings of CSCE Annual Conference, Hamilton, Ontario, 1990, Vol.IV, pp.183-193.
3. Akhras, G., and Fedoruk, J.A., "Trends of Expert Systems in Structural Engineering", Proceedings of CSCE Annual Conference, St.John's, Newfoundland, 1989, Vol.1A, pp.176-188.
4. Department of National Defence, "34 m Warren Type Wood Trusses and Columns Maintenance", Construction Engineering Technical Order, C-98-006-002/TP-002, National Defence Headquarters, Ottawa, Ontario, 1988.
5. Information Builders, Inc., "Level5 Object : User's Guide", Information Builders Inc., New York, 1990.
6. Maher, M.L., "Expert Systems for Civil Engineers : Technology and Application", American Society of Civil Engineers, New York, 1987, 148 p.
7. Microsoft Corporation, "Microsoft Windows : User's Guide", Microsoft Corporation Redmond, 1990, 640 p.
8. National Research Council of Canada, "National Building Codes of Canada, 1985", National Research Council of Canada, Ottawa, 1985, 454 p.
9. Shortcliffe, E., "Computer-Based Medical Consultation : MYCIN", American Elsevier, New York, 1976.

4th Symposium/Workshop

APPLICATIONS OF EXPERTS SYSTEMS IN DND

RMC, Kingston, Ontario, Canada

A HEURISTIC MULTIPLE TARGET TRACKER

Capt J.C.F. Beaupre[1], Dr M. Farooq[2] and J.M.J. Roy[3]

Abstract

Because of the proximity of sensor returns during crisis situations such as crossing tracks, standard multiple target tracking algorithms can generate unreliable target state estimates. The multiple hypotheses tracker (MHT) is a very powerful algorithm which does not generate a single track for each target, but rather forms hypotheses composed of possible tracks and associated probability values. The hypotheses tree generated can become quite large, and by extension, very demanding in computer resources. Often, the probability values of different hypotheses are close and deciding on which set of compatible hypotheses to display on the sensor scope is difficult to make until additional sensor data becomes available. Notwithstanding these limitations in the algorithm, the human operator must always be provided with timely and accurate visual information. Considering the difference between the behavioral patterns around an airport in peace-time and the aerobatics around a ship in war-time, we argue that the characteristics of physical targets provide useful information for deciding which hypotheses are the most likely. Considering the effect of rain on the performance of sensors, we argue that the environment in which the system operates also provides knowledge useful to tracking. Similarly, considering the effect of the density of targets on the algorithm itself, we argue that intrinsic characteristics of the tracking technique provide knowledge useful to its operation. These knowledge elements have been formalized as heuristic rules operating on the hypotheses tree of the MHT. In effect, the MHT algorithm has been modified to accept contextual and intrinsic knowledge in the form of heuristic rules.

[1]Graduate Student, [2]Professor, Dept of Electrical and Computer Engineering, RMC, Kingston
[3]Defence Scientist, Command and Control Division, DREV, Valcartier

INTRODUCTION

Motivation

Multiple target tracking (MTT) is the field of academic research which applies estimation theory to the tracking of multiple moving targets from sensor returns. Most MTT methods are algorithmic and can thus easily be implemented on digital computers. Many of the simple methods have long been implemented in working systems. Ever since the first prototype of the Automated Radar Terminal System (ARTS) was installed in Atlanta, Georgia, in 1964, civilian air traffic controllers have been provided with estimates of the kinematic features of the aircraft within their surveillance radar's field of view [12]. In recent years, the processing power of computers has greatly increased. Unfortunately, the implementation of many of the more demanding MTT algorithms into working systems is still not practical. Recent advances in software development techniques may provide the required environment for improving the performance of these algorithms. Expert systems are software programs which perform tasks that cannot easily be broken down into algorithms. Through this research, we are investigating the potential of applying recent developments in expert systems to MTT.

Related Work

A literature survey to review the goals, successes and failures of past research in applicable domain established the confines of the research topic and identified the different pertinent study areas within the academic context. As a result of this survey, we have situated our work under the intersection of the two academic fields of target tracking, and expert systems. A review of the printed literature on the subject has revealed the existence of a limited amount of research in this area.

Single Target Tracking (STT)

STT is a simplified target tracking problem whereby the kinematic states of a single target are estimated from noisy and cluttered sensor reports. Traditionally, the estimated track is obtained by building a linear model of the moving target and filtering the sensor returns through a standard estimation signal processing filter. The difficulty with STT lies with the formation of a system model which does not include any unknown input parameters such as pilot action. The literature survey revealed a single attempt at applying expert system technology to STT. Vanicola [15] proposes that an expert sys-

tem can be used for sorting out and selecting the best track amongst a set of different tracks obtained by different maneuvering filters. The driving knowledge base must contain the knowledge on the applicability of the different filters.

Multiple Target Tracking (MTT)

Most research falling under the MTT category deals with data association; that is process of matching sensor reports with tracks. Depending on the context in which a physical sensor operates and the characteristics of the sensor itself, a given sensor report can either belong to a tracked target, be caused by a new target, or be a false report caused by system noise. If the data association module performs its function properly, then the MTT problem simplifies to one of multiple STT systems operating in parallel. Typically, data association pairs are formed by creating a volume around the predicted position of an existing track and matching sensor reports falling within this volume to the track in question. This volume is referred to as the gate. It becomes more difficult to deal with the problem when more than one return falls within a given gate. A number of MTT algorithms have been developed to deal with such complex situations. Some of the more popular algorithms will be discussed in the second section. It should be noted at this point, though, that the application of expert system technology to the tracking of multiple targets is an active area of research.

Morawski [10], for one, advocates the use of heuristic rules for tracking a closed set of known moving objects which cannot easily be modelled by linear systems. An example of these objects are birds, and an example of an applicable rule is that "a duck is a bird that normally lands on water and never perches on branches". In this application, heuristic rules or rules of thumb, rather than gates, are used for matching sensor reports to targets.

Enhancement to more traditional solutions to the MTT problem have also been prcposed. Kountzeris [8], as an example, argues that an expert system could be built to combine and take advantage of the features of two MTT algorithmic methods. He proposes that the simplicity of the joint probability data association (JPDA) method could be combined with the adaptability of the track splitting filter (TSF) using a knowledge base as the link. Unfortunately, the reference falls short of discussing the implementation and the performance of the composite system. Reiner's [14] idea goes one step farther than Kountzeris. Reiner postulates that an expert system could manage several multiple target trackers held in an "algorithm inventory" and deployed by heuristic

rules which would select an algorithm according to a given goal function. Again , this system has not yet been implemented.

A third possible application of expert systems technology to MTT is one documented by Myler [11]. This system is quite unconventional in that it is not based on the estimation theory, but rather uses advances in imaging theory. Target tracks, in this system, are obtained by matching the sensor echos to pre-stored fuzzy tracks. The two modes of operation of this system include one for learning the fuzzy tracks and one for normal operation once the fuzzy tracks have been learned.

Other researchers in this domain promote the use of an expert system for controlling sensor resources such as output power, target illumination time and boresight shape according to derived situation assessments such as tracker performance, target density and electronic countermeasures (ECM). [1,15]

Multip. Sensor Fusion (MSF) and Higher Inferences

Another popular area of study is the assistance of expert systems to problems of MSF. The emphasis of such applications is often placed on track correlation; that is, matching of tracks obtained from individual sensors to form a comprehensive understanding of the environment [4,6,8,9,14]. Higher level inferences such as target classification, threat assessment and weapons-to-target assignment are often proposed as extensions to the track correlator. MSF and higher inferences are sometimes grouped in a category of expert system problems called "Command and Control" (C^2) problems [4,7,9,14,15].

Proposal

We propose a different approach for applying simple expert system tools to the MTT problem. Reliability is of prime importance during crisis situations such as crossing tracks, but because of the time and space proximity of the sensor reports, standard multiple target tracking algorithms can generate relatively unreliable target state estimates. The multiple hypotheses tracker (MHT) is a very powerful algorithm (and demanding in computer resources) which can handle difficult situations by differing the formulation of hard decisions. It does not generate a single track for each possible target, but rather forms hypothetical tracks with associated probability (or likelihood) values. Often, the probability values of different hypotheses are close and the decision of which set of compatible hypotheses to display on the sensor scope is difficult to make

until additional sensor data becomes available. Notwithstanding this limitation in the algorithm, the human operator must always be provided with timely accurate visual information.

Heuristic reasoning is a problem solving methodology associated with expert systems. It is sometimes applied to problems which cannot easily be solved using an algorithm. Applicable problems usually involve missing or uncertain information and are solved by traversing a decision tree using a heuristic inference (or reasoning) mechanism which attempts to duplicate the reasoning process of a human expert. [3]

We believe and can demonstrate that heuristics can be formulated to improve the performance of the MHT. These rules act on the tracks, hypotheses, and corresponding probability values to decide which hypotheses are the most representative of reality. Some of the rules we have designed assist in managing the hypotheses tree and, by extension, assist in managing computer resources. We have also been successful in detecting and following the possible manoeuvre of a target. In effect, we have modified the MHT algorithm to accept and process knowledge of the context (or environment) in which it operates and on its own strengths and weakness.

To evaluate the performance of our concept, we have built a prototype. The first step in building this prototype was deciding on the context in which our multiple target tracker will operate. Because we wanted to keep the contextual rules as simple as possible while demonstrating the usefulness of a working heuristic multiple target tracker, we decided to simulate the airborne environment of Canadian Forces Base (CFB) Portage, a small military flight training school of the Prairies as viewed through the returns of a modified area surveillance radar (ASR). This simulated ASR is untypical in that it provides the range, azimuth and elevation of airborne objects while a typical airfield ASR only provide range and azimuth information.

Outline

This document consists of three sections. Following this introduction, the second section expands on the academic fields of concern to the research. It describes contemporary MTT algorithms and discusses their strengths and weakness. It considers current methods of programming heuristic reasoning and outlines the implementation considerations. It also outlines the theory behind the MHT and proposes our concept for integrating the heuristic knowledge to the standard MHT. The third section describes our prototype based on the theory described in the previous section and compares the per-

formance of our heuristic tracker with the standard MHT. It also concludes this document and recommends future areas of research.

BACKGROUND THEORY

Introduction

The theoretical foundation of this work was obtained by studying recent published work in three areas. First, contemporary MTT methods were studied in order to understand the strengths and weakness of the different techniques and to choose the technique showing the most potential of integration with heuristic reasoning. Secondly, publications on the application of expert system tools to problems similar to MTT were reviewed. Lastly, to ensure the accuracy of the prototype simulator; information on the characteristics of small aircraft in flight was gathered, the applicable characteristics of ASRs were studied, the air traffic control (ATC) procedures were reviewed, and the features of visual and automated ATC support equipment were considered.

Before we can move on to discuss different techniques for following multiple moving targets from sensor reports, we must first get acquainted with estimation theory as it applies to the tracking of a single target.

Single Target Tracking

Figure 1 illustrates a simple estimation filter. It accepts noisy signals as its input and produces "clean" signals for the output. In a target tracking application, the noisy signals are the measured states of a target and the "clean" signals are its estimated states. Estimates for all of the target states can be produced provided the measured signals form an observable set of the target states. As an example; assuming that the range, azimuth and elevation measurements of a target are given and that the target moves with a constant velocity; then not only can we estimate the three dimensional location of the target, but we can also estimate its three dimensional speed and heading.

A target tracking filter will thus accept sensor measurements of a target and will estimate the kinematic features, or the track, of the target which caused these measurements. The filtering process normally entails simplifying assumptions to reduce the computational burden and to contend with the system's limited knowledge on the tracked object. In particular, a tracking filter estimating the states of an aircraft cannot

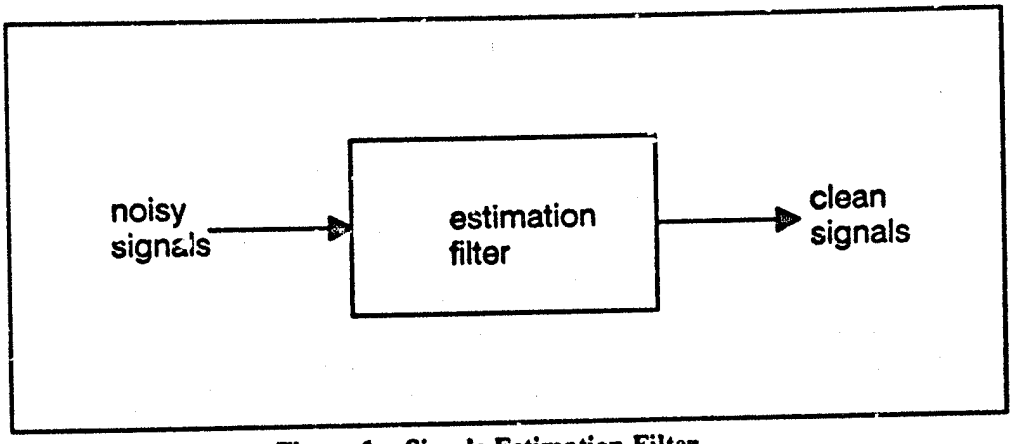


Figure 1 – Simple Estimation Filter

easily be provided with knowledge on the actions of the pilot. A common simplifying assumption made is that the moving object can be modelled by a set of linear constant coefficient ordinary n^{th} order differential equations. Most often, for computational reasons, "n" seldom exceeds two and is normally limited to one.

The Kalman filter is such a linear filter. It also has the important quality of being recursive so that a history of past measurements does not have to be maintained. The popularity of this filter is founded on the fact that it is an optimal recursive filter that will perform as well as can be expected of any other recursive filter as long as the physical target behaves according to the assumed model. In other words, given a first order model, the Kalman filter will behave optimally unless the target accelerates (or manoeuvres).

Multiple Target Tracking

Earlier we mention that most research work in MTT is directed toward data association. Data association, in the single sensor context, can be defined as the process of characterizing a given sensor report as either belonging to a known target, to a previously untracked target, or to a false alarm. Throughout this document sensor reports which do not correspond to any physical targets are called false alarms (FA).

Basically, three popular MTT techniques have been developed. The earliest and easiest to implement is the nearest-neighbor (NN) algorithm. The NN assumes that the best correlation is the one that associates tracks and reports that are closest, in some statistical sense. This algorithm, however, undergoes severe performance degradation as target and false alarm densities reach moderate levels. It forces an association which

in many cases is erroneous, and has no formalized capability to account for the fact that a target is not always detected.

The second technique we discuss is the Joint Probabilistic Data Association Filter (JPDAF) [2]. In this algorithm, all of the reports reasonably close to a given track are used to form a weighted average and the result is used to update the track. The weights are based on the probability that the report originated from the target and is a function of statistical distance. This algorithm has several drawbacks. While it shows the ability to maintain continuity superior to the nearest-neighbor algorithm, it does this at the expense of accuracy. This is apparent since all reports are used in computing the track update even though at most one report is really from the target. The others must be reports from other targets or from false alarms. Another limitation to both of these techniques is that track initiation and track termination are not inherent to the algorithms, but must be handled separately.

The third and last candidate MTT technique is the Multiple Hypothesis Tracker (MHT) [2,13]. Here, instead of trying to resolve a difficult association immediately, all possibilities are enumerated as hypotheses. The probability of each hypothesis is computed, and ideally, all hypotheses are maintained until there is enough data to make a decision as to the correct hypothesis. Since it would be extremely computationally demanding to maintain all hypotheses, a suboptimal implementation which includes a mechanism for pruning unlikely hypothesis must be used. This algorithm contains a number of advantages over the other two techniques. Some of these advantages are that track initiation is an integral part of the algorithm, the manoeuvre detection can be handled within the basic algorithm, and last but not least, one of the hypotheses must be the correct one.

A useful concept for MTT is that of gating [2]. Gates, or correlation gates, are sets of positional and velocity limits in each coordinate. A sensor report which falls within the gate of a track is said to correlate with that track. It should be noted that all of the three techniques of MTT described above support gating, but they can also be implemented without it.

For illustrative purposes, we will now consider a simple data association problem and discuss the process followed by each of the three algorithms introduced above. The situation considered is shown in figure 2. Our system, which uses gating, has been tracking targets for some time. Two tracks, T1 and T2, are being followed. We now stand

at the k^{th} sample (k is a positive integer). A gate is formed around each of the two tracks and is drawn as an oval centered on the predicted positions ($T1_{k|k-1}$ and $T2_{k|k-1}$) for sample k given the measurements up to sample (k–1) . Three measurements (or reports), M_a, M_b and M_c, fall within the boundaries of either of the two gates. All three measurements fall within the gate of T1 while only M_a and M_b correlate with T2. A fourth measurement, M_d, does not correlate with any of the two tracks .

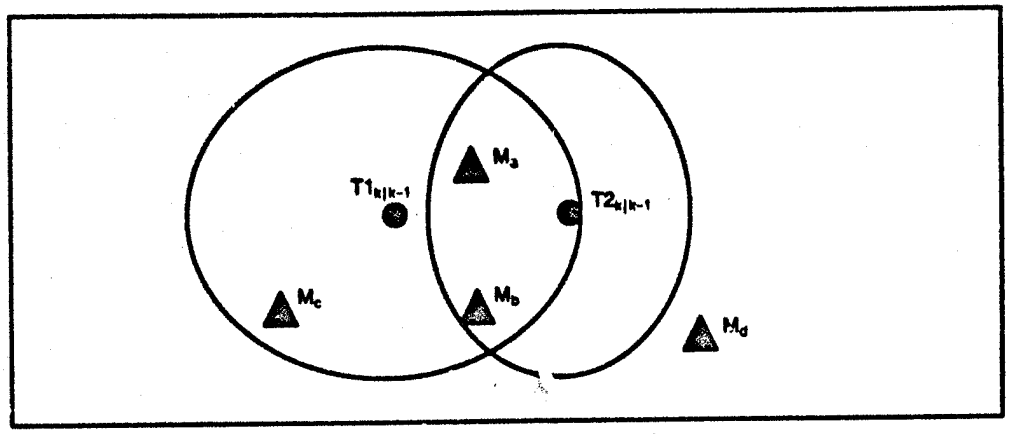


Figure 2 – Data Association Problem

Let us define the residue to be the distance between the predicted position of a track and the location of a given sensor report. The first algorithm, the NN, processes T1 by calculating the residue between the position labeled $T1_{k|k-1}$ and the locations of M_a, M_b and M_c and uses the closest measurement to update the track. Similarly, for T2, it uses either M_a and M_b depending on which one is closest to the location labeled $T2_{k|k-1}$. Notice that no constraints, within the standard algorithm, prevents M_a from updating both tracks. Also, the basic algorithm, implemented without gating, cannot deal directly with missed reports, that is physical targets which are not reported by the sensor. There exist variances to this algorithm, one of these being the Track Splitting Filter (TSF) which attempts to deal with the problem of multiple measurements falling within a gate.

The next technique we introduced is the JPDAF. Again, as with the NN, the residues between compatible tracks and measurements are computed. To update track T1, a hypothetical measurement is formed by calculating a weighted average of residues between T1 and each of the three measurements falling within its gate. The weights used are based on the residues themselves. The smaller the residue, the higher the weight.

Track T2 is also updated through the same process using M_a and M_b. As mentioned earlier, the JPDAF is not as accurate as the NN. It disregards the undebatable premise that at most one measurement a scan is caused by physical target. On the other hand, it is more capable of maintaining continuity since its performance will suffer less than the NN whenever the wrong measurement happens to be closest to a predicted target position.

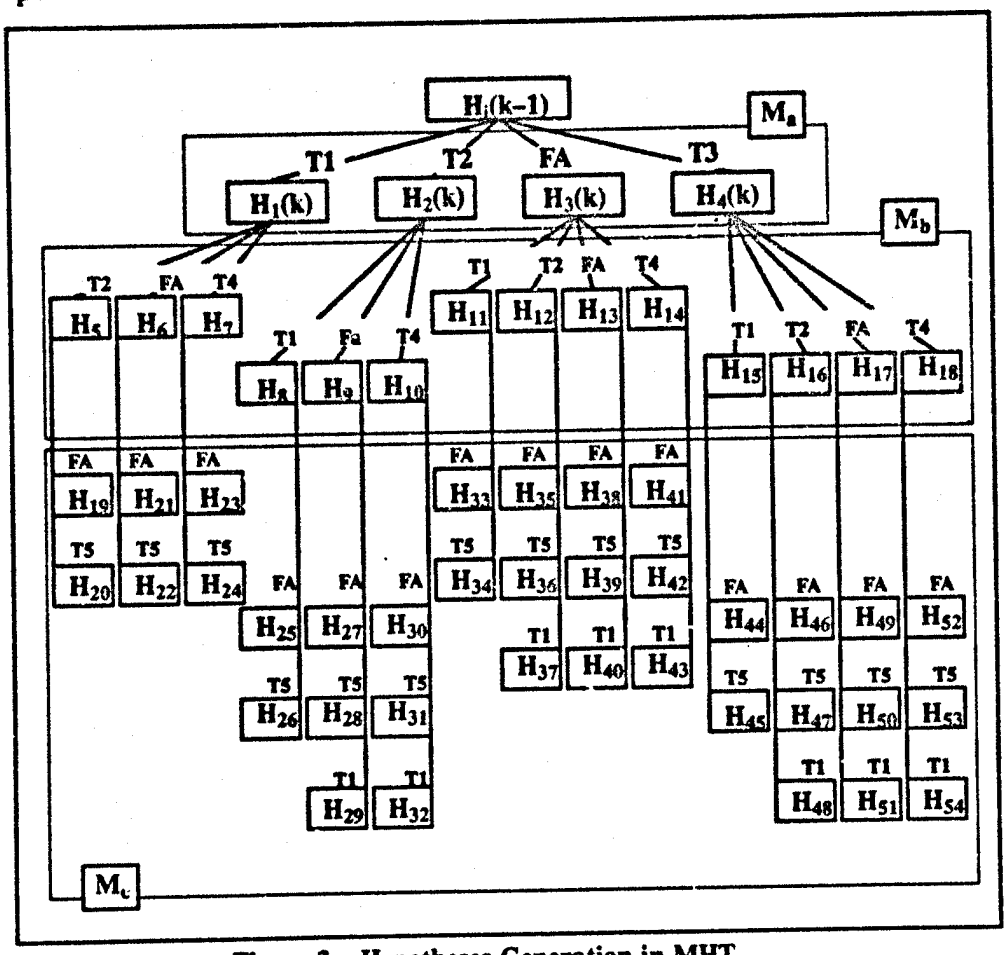


Figure 3 – Hypotheses Generation in MHT

Consider again our example. Figure 3 shows part of the hypotheses tree that would be generated by the MHT. Suppose an earlier hypothesis at scan k–1, identified in the figure as $H_i(k-1)$, included the existence of the two tracks T1 and T2. Because M_a correlates with both tracks, four child hypotheses are created from $H_i(k-1)$. The first

($H_1(k)$) assigns M_a to T1; the second ($H_2(k)$) matches it to T2; the third hypotheses ($H_3(k)$) labels M_a as being a false alarm (FA); while the last one ($H_4(k)$) assigns it to a new target it calls T3. Now, M_b also falls within the gate of both targets. Each of the four new hypotheses formed by processing M_a generate additional hypotheses when M_b is processed. Given that M_a was assigned to T1 ($H_1(k)$), three new hypotheses, $H_5(k)$ through $H_7(k)$, are generated. Note that M_b cannot be assigned to T1 since T1 has already been updated under its parent hypotheses. M_b, on the other hand, can still be used to update T2 (H5(k)). It could also have been caused by system noise ($H_6(k)$) or it could indicate the existence of a new target called T4 ($H_7(k)$). On the other hand, assuming $H_2(k)$ to be right, three new hypotheses are formed and join the tree under $H_2(k)$. Similarly, four new hypotheses are generated under each of $H_3(k)$ and $H_4(k)$. In effect after processing two measurements of scan k, our initial hypotheses $H_i(k-1)$ gave birth to 14 child hypotheses. After processing the third measurement (M_c) in a similar manner, the number of hypotheses increases to 36. For lack of space, figure 3 does not show the new leafs created by processing M_d. Because M_d does not correlate with any existing track, only two hypotheses are generated under each of the hypotheses formed by processing M_c. The first one characterizes M_d as a new target (T6) while the second one makes it a false alarm. The number of end hypotheses, thus, doubles to 72. It should also be noted that the algorithm provides means for calculating the probability (or likelihood) of each hypothesis. This probability value is a function of the probability of its parent, the expected density of real targets, the expected density of false alarms, the expected probability of detection, the residue as defined earlier, the expected quality of the tracks (track covariance matrix) and the expected precision of the measurements (measurement covariance matrix).

The fundamental difference between the MHT and the other two algorithms described above is that it is measurement oriented rather than being track oriented. Track oriented algorithms require prior knowledge of the existence of targets before they can form estimates of kinematic features. They form track to measurement associations according to the probability that a given measurement belongs to a known track. Such techniques, i.e. the JPDAF and the NN, do not formally provide means of recognizing the existence of a new target or the termination of a vanished target. Measurement oriented techniques, such as the MHT, use the prior knowledge that we have a report, and compute the probability that this report belongs to an existing target, to a new target, or is caused by a false alarm. Clearly, techniques based on the track oriented model

are more manageable since the measurement oriented model involves excessive growth of hypotheses that must be stored and manipulated.

Nevertheless, with the computation power of present day digital computers, and using some simplifying assumptions based on the environment (or context) on which the MTT operates, an MHT can be fine tuned to perform adequately. Knowledge engineers find the algorithm quite attractive in that it generates a multitude of mutually exclusive interpretations. The challenge resides in choosing the most likely interpretations to be displayed to the human operator. Also, unlikely interpretations can be removed from the hypotheses tree to limit its growth and maintain the computer resources requirement within manageable limits. Again, the problem is in establishing which interpretations are unlikely.

Heuristic Reasoning

Expert systems, in this document, are systems which attempt to model the heuristics or rules by which human experts solve problems. Human experts can reason with incomplete, uncertain and contradictory knowledge. Human faces are all different, but a baby girl easily recognizes her mother's face very early in life even though she has assimilated and can understand few facts about her environment. In effect, this child is an expert at recognizing human faces. In fact, she is probably more efficient in this role than any of the contemporary artificial systems. Human experts commonly apply their knowledge heuristically. They explain their decision making process by using analogies, examples, guesses, or rules that apply within restrictive conditions. Attempts to automate such decision making processes often perform poorly because of the difficulty in formalizing all of the necessary rules. This problem is being looked into by researchers in the field of knowledge acquisition.

Let us consider three classes of problem solving methodologies and define a two dimensional space we call the solution space. Within this space, a problem solving methodology is described as a search tree. The root represents the initial states of the problem, branches describe applicable interim solutions and the end nodes characterize possible complete solutions to the problem. Figure 4 illustrates the three classes we will describe.

The first class, figure 4(a) the exhaustive search, is one where all of the possible solutions are investigated. The tree is traversed either in a depth first or a breath first manner and admissible branches are never disregarded. Comparing figures 3 and 4(a),

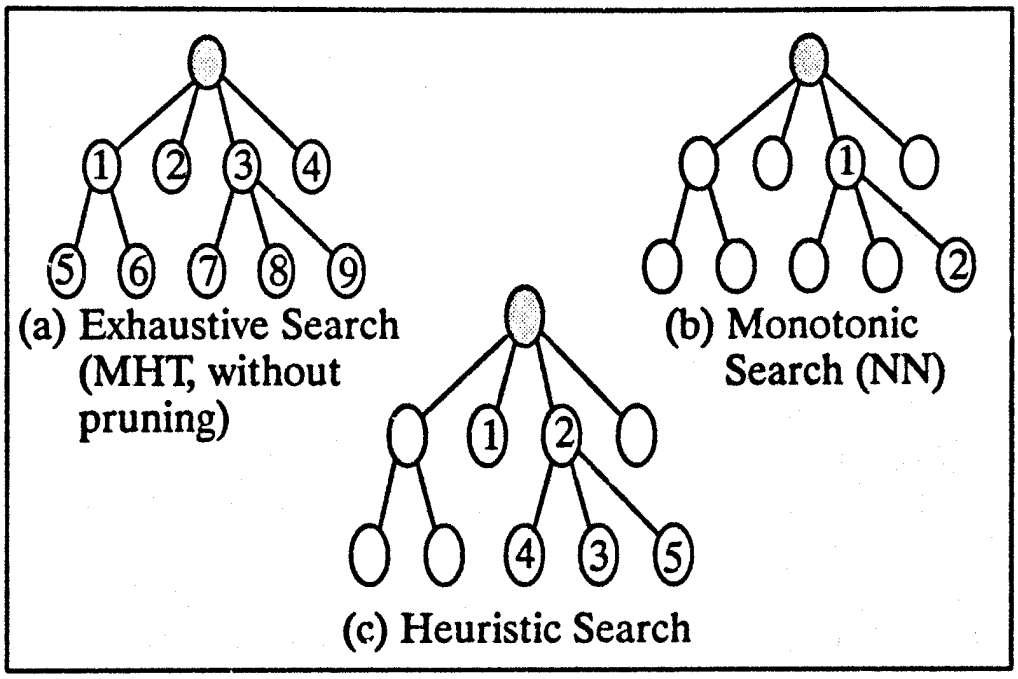


Figure 4 – Searching for a Solution

we quickly realize that the MHT, without pruning, falls within this category. An important consideration in implementing such problem solving methods is that decisions are never firm, but are continuously reviewed. A problem with this technique is that it provides a variety of possible solutions, but no way of distinguishing which solutions are right or the best.

The second class, illustrated in figure 4(b), includes techniques, such as the NN and the JPDA, where hard decisions are formed at every level of the tree so that, once the decision is formalized, it can never be reviewed. We refer to this problem solving methodology as monotonic reasoning with reference to the impossibility to review decisions made earlier in time.

In some ways, the third and last class, shown in figure 4(c), is a compromise between the other two. The intent is to improve the manageability of the exhaustive search by eliminating unlikely, but admissible, solutions. Hard decisions, similar to the ones made in monotonic reasoning, are made to distinguish between likely and unlikely nodes. Nodes labelled as unlikely are removed (or pruned out) from the decision tree. Soft decisions are also continuously reviewed to establish which of the possible end solu-

tions are good and worth returning to the human operator (or to other systems). These decisions are guided by the amount of computer resources available, the timeliness requirements of the solution, and the acceptable error or impreciseness rate. The conditions (or rules) on which these decisions are based are determined by the environment in which the system operates and on the intrinsic characteristics of the algorithm. These conditions are also often subjective and formed by the designer according to his rationalization of the environment and his comprehension of the algorithm. It is the subjective nature of these conditions that characterizes this reasoning class, which we call heuristic reasoning.

We have been introduced earlier to the problem of potentially unmanageable (NP hard) growth of the MHT hypotheses tree. Obviously, the MHT cannot be implemented directly as an exhaustive search problem. Not only would the computer resources quickly extend above manageable limits, but so many possible solutions would exist that it would be impossible to find the correct one. Unlikely hypotheses must, then, be pruned out to keep the problem manageable (NP complete). A method for confirming good hypotheses must also be put into place to provide the means of returning the human operator with the system's best interpretations of the sensor's environment. These changes to the standard MHT would make it fall in the heuristic class of algorithms.

The difficulty now lies in establishing the criteria to use in characterizing hypotheses as good, likely or unlikely. Obviously, the easiest method to distinguish unlikely hypotheses from the others is to use their probability values. We can either remove the hypotheses that fall under a probability threshold or keep the n (n is a positive integer) most probable ones. Good hypotheses can also be defined as the ones of highest probability. Chen [5] proposes an efficient method of implementing such a selection technique based on the branch–and–bound algorithm. Using his technique, the computer would find the good and likely hypotheses without formally forming them. Unlikely hypotheses would then never be created. Another advantage of the branch–and–bound algorithm is that the selection process could be implemented on a multiprocessor system.

The problem with this technique is that the probability value of hypotheses is based on features which are assumed and may not be representative of reality. As an example, as weather changes, the density of false alarms is normally affected; but the manner in which it is affected is unpredictable. As a result, the expected density of false

alarm is never well known, but this value is necessary for the computation of the probability of every hypothesis which includes the supposed existence of a false target. Our attempt is to find heuristic ways of characterizing hypotheses using the knowledge of the environment in which the sensor operates, the knowledge of the characteristics of the targets normally found in view of the sensor and the knowledge of the intrinsic characteristics of the algorithm.

THE HEURISTIC MTT PROTOTYPE

Introduction

Figure 5 shows the block diagram of the heuristic based MTT built to investigate the viability of our concept. The MTT itself is depicted by the center block in the figure. It is composed of two sub–modules. The first sub–module is a standard MHT while the second formalizes the concepts described in the last section. We have also designed a scenario generator which allows us to build suitable flight trajectories and set up scenarios by combining different trajectories in time and space. These scenarios are then used by the sensor data generation module which generates simulated sensor reports which are then forwarded to our tracker. Our last module is our graphical display module which accepts the system's best interpretation of reality and displays the corresponding tracks. Many of the heuristic features of our system can easily be removed so that we can compare their effect on the performance of the overall system.

Physical Environment of the Prototype

The immediate airspace around an airport is strictly regulated. To maintain air traffic separation, pilots can either use the services of air traffic controllers or obey visual flight regulations (VFR). The choice between these two alternatives depends on the intentions of the pilot and on the weather. As an example, to land or take–off, pilots must always contact the controllers. They may not need any clearance, though, if they are already airborne and simply wish to fly through controlled airspace without interfering with other traffic. In either event, there exist strict guidelines for vertical and horizontal separation between traffic, restrictions on the altitude of flight depending on heading, and formalized standard flight behavior patterns which reveal much about the intention of the pilot. As a result of these regulations, the behavior of targets within the airspace of an airport under normal circumstances, is somewhat predictable [12].

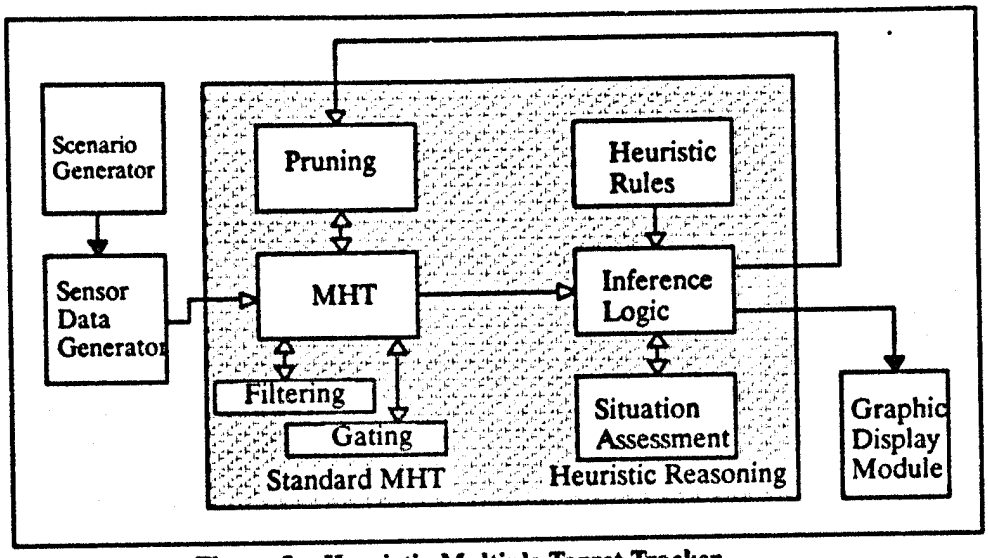


Figure 5 – Heuristic Multiple Target Tracker

Knowledge based on the predictable features of targets can assist in forming some of the heuristic rules necessary in improving the manageability of the MHT.

The type of traffic around a given airport also tends to be predictable. As a test location, we have studied the airborne environment of a small military training airport of the Canadian prairies called CFB Portage. Very rarely does any traffic other than the training aircraft venture within the controlled airspace of Portage. Accordingly, characteristics of expected targets such as cruising speed, climb and descent rates, and maneuverability can be used to form restrictions on the characteristic of the tracked targets. In turn, these restrictions may assist in characterizing hypotheses as good, likely or unlikely.

We have developed different scenarios to test our prototype. Our principle scenario lasts 25 minutes and includes up to nine targets of various kinds performing aerobatics typical to the Portage airspace. Seven of these targets (three Tutor jets, three Musketeer propellor planes and one Jet Ranger helicopter) are common to the Portage environment, the eighth one is a 707 flying through the scan volume of our sensor at a high altitude, and our last aircraft is a civilian Cessna 150 which has similar flight characteristics to the military propellor trainer, the Musketeer.

The sensor, its location and its characteristics can also play a role in forming the heuristic rules. If a radar is located in such a way that the ATC tower shadows targets

behind it, the probability of receiving an echo from this shadow is low. The knowledge such as this can be of assistance to the MTT. Similarly, weather and other environmental elements (such as sea clutter) is known to affect the performance of sensors. The effect of these elements, we believe, should also be integrated in a working system.

Our prototype uses a simple model for the sensor. It simulates a modified Area Surveillance Radar (ASR) with a scan range of 40 km and a azimuth range of 360°. The ASR of a typical airfield provides range and azimuth information of aircraft within its field view but our simulated sensor also returns elevation information. In fact, our simulated sensor has an elevation range of 0° to 60°. We justify this inconsistency between our simulated sensor and physical systems by noting that the tracking algorithm of most working systems use the altitude provided by the Secondary Surveillance Radar (SSR) transponder of physical aircraft and are thus able to perform three dimensional tracking. Because we initially restricted the scope of our work to the single sensor case, we felt our approach of simulating a three dimensional ASR reasonable.

Performance of the Prototype

Because of the responsibility resting on the shoulders of air traffic controllers, their support tools must be extremely dependable. Accuracy, timeliness, and completeness of information is crucial.

We have thus chosen to display a large number of tracks. Fortunately, because most of the tracks are concentrated around the location of physical targets and many are so close that they overlap, we found that this practice did not overly clutter the operator scope. Another heuristic is that the association of a measurement to an existing target or to a new target is always favored over the characterization of this measurement as a false alarms. This has improved the timeliness of the algorithm while rendering the algorithm more conservative in its measurement to track correlation.

We have also been successful in using target related features in forming heuristic rules. Tracks which are too high to belong to any expected physical targets are weeded out. We also reduce the number of possible correlations by preventing gates from having radii larger than the distance that can be travelled by targets flying at a multiple (between two and three) of the maximum speed of aircraft expected within the surveillance volume. Because new tracks have high covariance, the correlation gate formed in the classical manner around a new track is quite large. This reduction in gate size has improved the timeliness of the system without affecting the accuracy. Another use for these

expected maximum speed parameters is in initializing the velocity elements of the track covariance matrix.

Unfortunately, because the versatility of our sensor model is limited, our prototype does not allow us to observe the effect heuristics on changing environmental features. In our sensor model, the probability of detection of physical targets is kept constant and is independent of the range and the radar cross section of individual targets. Also, the density of false alarms and the standard deviation of measured positions are constant in time and space. These parameters must be set before running the sensor module and cannot be changed as the scenario unfolds. Accordingly, heuristic rules based on changing environmental features (such as weather and shadowed targets which affect these parameters) cannot be tested with our present prototype. We have been successful, though, in projecting the tracks of non–maneuvering targets leaving the surveillance volume (elevation above 60°) and in resuming normal tracking as they re–enter the viewed space.

The changes mentioned above are minor, but have revealed themselves to be effective in the air traffic control environment. Because of the regulations that targets follow in such an environment, they are normally well behaved and, thus, easy to track. Our main scenario of 25 minutes (300 scans) involves nine targets behaving within regulated directives. We have been successful in displaying timely, accurate, and dependable information. The tracking module of our system normally returns the estimated interpretation of reality within three second of processing time, it very seldom looses track of a target even if the target is maneuvering, it displays all new physical targets within four scan periods and it seldom shows false targets. We have also been successful in associating tracks belonging to the same target and are now in the process of developing heuristics for situation assessment. The situation assessment module will be responsible for finding tracks belonging to physical targets that do not comply with the behavior expected of them. Examples of such unauthorized behavior are: aircraft flying too low, aircraft with conflicting headings, and unauthorized separation between two aircraft.

ACKNOWLEDGEMENTS

We wish to acknowledge the much appreciated contributions provided to this research work by a number of individuals from RMC. L. de Kok has developed single handedly the complete graphics package without which this prototype would be of little

impact. M. Nedvidek has been very helpful with his advice on the programming aspects of the work. S. Lim, S. Bruder, and T. Quach have also been very useful for discussions on theoretical matters. These individuals are all members of a project group funded by the Defence Research Establishment Valcartier and they are in the process of developing a multiple sensor, multiple target tracking simulator called the "Concept Analysis and Simulation Environment for Automatic Target Tracking and Identification" (CASE-ATTI) system.

REFERENCES

1. Addison, E.R., "Design Issues for a Knowledge Based Controller for a Track-While-Scan Radar System", Proceedings of the SPIE - The International Society for Optical Engineering, vol. 635, Orlando, FL, USA, 1986, pp. 276-279.

2. Blackman, S.S., Multi-Target Tracking with Radar Applications, Norwood, MA, USA: Artech House Inc., 1986.

3. Charniak, E. and McDermott, D., Introduction to Artificial Intelligence, Reading, MS, USA: Addison-Wesley Publishing Company, 1987, p. 261.

4. Chaudhuri, S.P. and Agrawal, R. "Artificial Intelligence Applications to Command, Control, and Communications Systems/Subsystems", Proceedings of the SPIE - The International Society for Optical Engineering, vol. 1100, Orlando, FL, USA, 1989, pp. 85-96.

5. Chen, C.W., Walker, R.A., Feng, C.H., "A Branch-and-Bound Algorithm for Multiple-Target Tracking and its Parallel Implementation", Proceedings of the 1988 American Control Conference, vol. 3, Green Valley, AZ, USA, 1988, pp. 359-364.

6. Engelmore, R. and Morgan T, "Recent Applications (1981-1985)", Blackboard Systems, Ed. R. Engelmore and T. Morgan, Workingham, England, U.K.: Addison-Wesley Publishing Company, 1988, pp. 349-352.

7. Fahmy, A.M. and Titus, H.A. "Horizontal Estimation for the Solution of Multitarget Multisensor Tracking Problems", Proceedings of the 1987 American Control Conference, vol. 3, Green Valley, AZ, USA, 1987 pp. 2103-2108.

8. Kountzeris, A. Towards an Intelligent Multiple Target Tracker, 1989.

9. Llinas, J. "Toward the Utilization of Certain Elements of AI Technology for Multi-Sensor Fusion", Applications of AI to Command and Control Systems, Ed. C.J. Harris, London, England, U.K.: Peter Peregrinus Ltd, 1988, pp. 37–54.

10. Morawski, P., "A Hybrid Approach to Target Tracking", Proceedings of the Annual AI Systems in Government Conference, Washington DC, USA, 1989, pp. 80–87.

11. Myler, H.R., Thompson, W.E. and Flachs, G.M., "Application of Expert System Techniques to a Visual Tracker", Proceedings of the SPIE – The International Society for Optical Engineering, vol. 548, Arlington, VA, USA, 1985, pp. 124–127.

12. Nolan, M.S., Fundamentals of Air Traffic Control, Belmont, CA, USA: Wadsworth Publishing Company, 1990.

13. Reid, D.B., "An Algorithm for Tracking Multiple Targets", IEEE Transactions on Automatic Control, vol. AC-24, Dec 1979, pp. 843–854.

14. Reiner, J., "Application of Expert Systems to Sensor Fusion", IEEE National Aerospace Electronics Conference, USA, 1985, pp. 1444–1450.

15. Vannicola, V.C., and Mineo, J.A., "Applications of Knowledge Based Systems to Surveillance", Proceedings of the 1988 IEEE National Radar Conference, New York, NY, USA, 1988, pp. 157–164.